COMPUTER AIDED DESIGN
OF
MODULAR FIXTURE ASSEMBLY

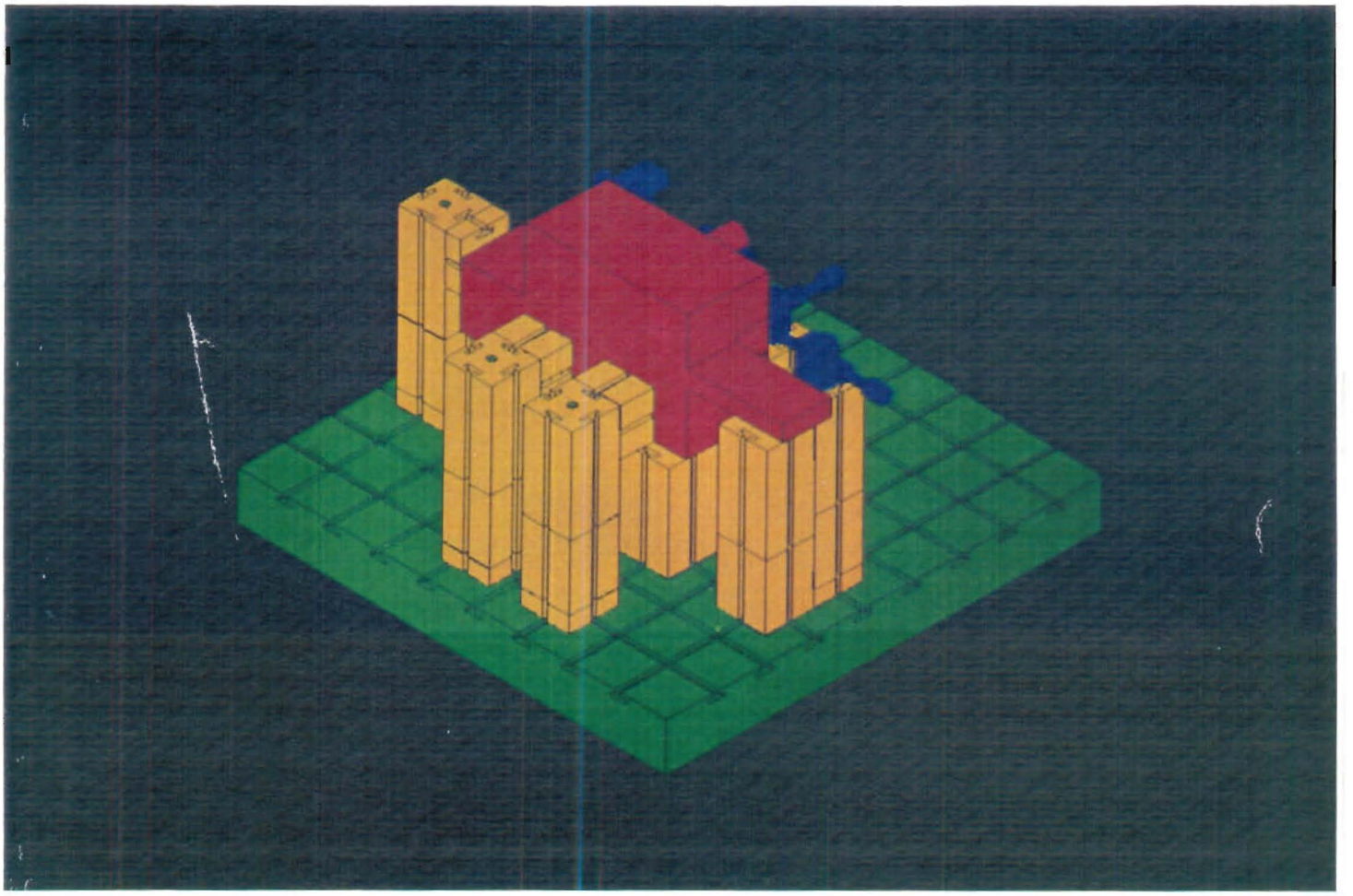
This Thesis is submitted for the Degree of Doctor of Philosophy
in Mechanical Engineering at the University of Canterbury,
Christchurch, New Zealand.

by

Bryan Ngoi Kok Ann

University of Canterbury

1990



To my wife

Jennie

for her encouragement and support

CONTENTS

ACKNOWLEDGEMENTS

I am grateful to my supervisor, Dr K Whybrew, for his patient guidance, criticism, and confidence.

I am also thankful to Professor H McCallion and Dr R J Astley for their interest in my work.

I would also like to extend gratitude to Mrs J Vogan for her valuable help and encouragement.

Last but not least, I would like to express my thanks to all my fellow students and the staffs of the Mechanical Engineering department for their help and support.

ABSTRACT

The design and construction of fixtures is a major hindrance to the reconfigurability of flexible manufacturing systems. A promising method is to use a modular fixturing system. There remains, however, a considerable problem in the design of appropriate assemblies of such fixturing elements.

This project describes a program for the automatic design of assemblies of modular fixturing elements. Its input is a description of the workpiece geometry, the machining envelope and the fixturing points. Its output is an automatically generated fixture design. The program incorporates systematic design procedure in conjunction with a large library of rules governing the movement and combination of the modular elements.

A key feature of the system is the development of a suitable spatial representation technique, to describe those regions of space which are occupied by the workpiece itself, by the fixture and by the tools used to produce it. This spatial representation system permits the program to search and identify objects and object intersections. It is also able to determine the relative position of the modular elements during the design process.

A design example was given to illustrate the method.

SUMMARY

Chapter 1 describes how fixtures restricts FMS in their long term flexibility, or reconfigurability. Modular fixturing systems were identified as being the most promising means of improving reconfigurability, and computer aided design of modular fixture assembly further increases the reconfigurability of the fixture. This chapter ended with the objective of the project.

Chapter 2 gives a comprehensive review of the fundamental principles of fixture design and the role of fixture design in process planning.

Chapter 3 describes the various aspects of computer aided design of modular fixtures, and reviews the relevant literatures at the end of each appropriate section. It then outlines the method used to automate the design of the modular fixture assembly. The scope of the project is also included in this chapter.

Having studied the available modular fixturing systems, the design procedure for the modular fixture assembly has been rationalised and is described systematically in chapter 4, 5, 6 and 7.

Chapter 4 introduces the fundamental concepts of the method. It categorised the modular fixturing elements into three groups:

(a) Locators, clamps and supports

(b) The baseplate

(c) Modular blocks and shims

The design of the modular fixture assembly essentially involves selecting a feasible combination of the modular blocks and shims to bridge the gap between the locators, clamps, and supports, and the baseplate. For efficient design procedure, the concept of fundamental structures is introduced. From the concept of fundamental structures, the following concepts are further developed :

(a) concept of towers and mounting blocks
(b) concept of direct and indirect mountings

Chapter 5 outlines the overall structure of the design procedure. It gives a "bird eye" view of the design procedure. The input to the program consists of a description of the workpiece geometry, the machining envelope and the fixturing points. The program automatically generates and combines 'towers' of modular blocks. The combination of towers are governed by rules stored in the rulebase. These towers are then systematically mounted onto the baseplate to form the required fixture assembly. The details of the various stages of the design procedure is given in chapter 6 and chapter 7.

Chapter 6 describes the tower generation program. The tower generation program considers each locator, clamp, or support, and builds appropriate 'towers' of modular blocks to bridge the gap between the reference point on the locator, clamp, or support and the baseplate. If any intersection occurs, the program will consult the rule-base for appropriate actions to be taken.

Chapter 7 describes the tower mounting program. The towers generated by the tower generation program are standing on the baseplate unmounted. The tower mounting program has to mount these towers onto the baseplate. Some towers are mounted directly and some indirectly. The objective of the tower mounting program is to mount these towers as directly as possible.

Chapter 8 is devoted to the spatial representation system. It describes the spatial matrix representation of a general object. It then outlines the special numbering system and explains how the numbering system enables the program to identify intersecting objects during an intersection. Finally it describes two important utility programs, for manipulating objects, with illustrations. The procedure for developing these utility programs and some useful tools are also outlined in the this chapter.

Chapter 9 describes the structure of the system programs. It also explains how the modular programming technique and the dynamic variables are employed to fully utilise the limited memory.

Chapter 10 presents eleven design examples using the program. The aim of this chapter is to illustrate the method.

Chapter 11 concludes the project.

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

1.1.1 FMS - pallets and fixtures

A typical flexible manufacturing system (FMS) is a group of integrated manufacturing cells, (usually numerically controlled machines), which are connected by an automated workpiece handling system and operated under computer control [1,2,3,4,5,6].

Pallets and fixtures are used to transfer workpiece from machine to machine. A pallet is essentially a precision plate which is able to hold the fixture, and has location features in itself, which enables it to be repeatedly and accurately positioned on a machine tool.

In order not to tie the machine up in unproductive time for loading and unloading workpieces, i.e. to avoid the non productive part of the cycle, the workpieces are pre-loaded in fixtures on pallets. Generally speaking, the loading and unloading is done off-line, and a common way of doing it is to mount the fixture on the pallet, and the workpiece is then clamped in place onto the fixture.

1.1.2 Flexibility and reconfigurability

The concepts of flexibility and reconfigurability have been examined by D.J.Williams [2] and are defined as follows:

"**Flexibility**, (sometimes short-term flexibility), refers to the ability of a manufacturing system to process a number of different parts from a pre-defined group of parts. For example, a system that makes a hundred different parts is more flexible than a system that makes twenty different parts. Manufacturing systems, even the most programmable, are usually designed to produce a pre-determined group of parts."

"**Reconfigurability**, (sometimes long-term flexibility), can be taken to refer to the ability of a manufacturing system to process a group of parts other than those for which it was designed, or which involve a significant product changeover. This measures how difficult is it to re-tool and re-program the system to make a different set of parts."

When a batch of workpieces are introduced to the FMS, it usually requires processing through several different manufacturing cells. Each production machine needs to change its programs and toolings with the least delay. This can be achieved on both programming and cutting tools, however this is not always true in the case of a fixture.

In FMS, fixtures are normally designed so that they can adapt to slightly different configurations within a part family, and several different adaptable fixtures may be required to handle a range of part families. Some examples of adaptable fixtures are given in section 3.2.1.

Every time a new component is introduced, (never manufactured before), the fixture has to be changed on the pallet. When a workpiece does not belong to any of the part families which the adaptable fixtures

are designed for, then it cannot be held by the existing adaptable fixture. A new fixture is required to hold such a workpiece.

The lead time required to design and manufacture the new fixture, restricts FMS in their long term flexibility, or reconfigurability. Poor reconfigurability means that it may often take several months lead time before the new component can be manufactured. For this reason, almost all FMS are captive manufacturing facilities, providing in-house manufacture of frequently repeated batches of well defined components. Attempts to use them for single batch jobbing manufacture are usually not successful.

To improve reconfigurability, fixture systems, which can be quickly rebuilt or adapted to accommodate new components, are required. There have been various attempt to increase reconfigurability by design of flexible fixtures, and these are describe in section 3.2.2.

Modular fixturing systems were identified as being the most promising means of providing such increased reconfigurability, and the direction of the work was based on modular fixtures.

1.1.3 Modular fixtures

A modular fixture generally consists of a large number of precise modular elements, which can be assembled manually in different configurations, to provide the location and clamping features needed for machining operations. Modular fixturing system can be assembled quickly and are reusable.

One such system is produced by the Chinese Aeronautical Technology Import/Export Company (CATIC). All elements in this system are keyed together using high tensile steel 'T-bolts', and mounted onto a baseplate. Figure 1 shows an example of a CATIC modular fixture.

1.2 STATEMENT OF PROBLEM

Modular fixtures can increase reconfigurability, but there remains the problem of fast and accurate design of the combination of modular elements. From a postal survey of some U.K. CATIC users, the following problems have been recognised.

1 The design of the modular fixture body lacks systematic procedure. It is currently based on experience and 'trial and error'. The designer's aim is to obtain a feasible combination of modular elements while satisfying the fixturing requirements.

2 The design process is slow and even experienced fixture designers requires several hours to complete. This problem has also been identified by Woodwark and Graham [34], Gandhi and Thompson [26] and Lewis [52].

3 There is no obvious method of recording fixture assembly details for future reference. According to Woodwark and Graham [34], this often leads to fixtures not being dismantled after use. In this case, any cost advantage is reversed, as an

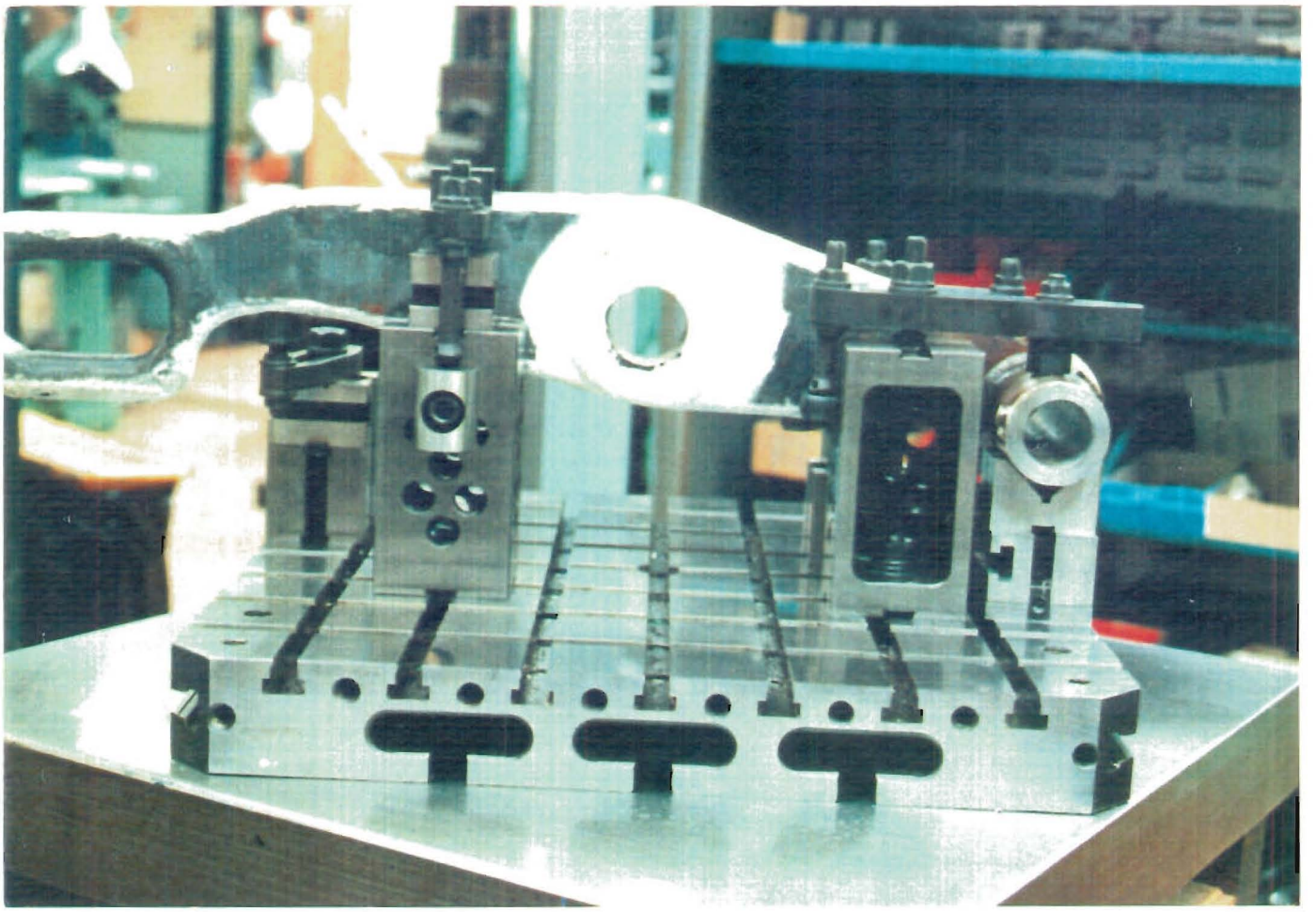


Figure 1 An example of the CATIC modular fixture

expensive versatile kit becomes another piece of dedicated equipment.

1.3 OBJECTIVE

The automated design of modular fixtures is currently in an embryonic stage of development. Fully automated systems are not yet commercially available, however, this is the subject of vigorous research on an international level.

The objective of this project is to provide a computer aided method of designing the fixture body, using a set standard modular fixture elements. The input to the design system consists of the workpiece envelope, the machining envelope, and the type and placement points of the locators, clamps, and supports. (It is assumed that all these have been determined prior to embarking upon this stage of the design). It is important to note that the relationship between locators, clamps and supports, and the component, is a fixed input to the system. This system takes this input and builds a structure to connect the fixed points on the input to a baseplate. The structure, which consists of combinations of blocks and shims, and the baseplate, is the fixture body.

CHAPTER 2

FIXTURES

The fundamental principles of fixture design described in section 2.1, and the role of fixture design in process planning described in section 2.2, are based on Eary and Johnson [7].

2.1 FIXTURE FUNDAMENTAL

The function of a jig, or fixture, is to locate and hold a component firmly in position during a manufacturing process such as machining, assembly or inspection and to support it by providing reaction to the machining forces. Jigs perform the additional task of guiding and supporting the tool, and are used, for example, in drilling operations. This additional task of guiding and supporting the tool is not required for NC machining, therefore, jigs are not applicable to the flexible manufacturing system.

A fixturing system consists of three basic types of elements: locators, clamps, and supports. Locators are used to position the workpiece in a state of static equilibrium, depriving the workpiece of its six degrees of freedom. Clamps are for pressing the workpiece firmly against the locators and holding it there against the action of cutting forces. In addition to locators, supports are sometimes needed to control deflection of the workpiece due to its own weight, tool forces and clamping forces.

The fixture must accomplish the following objectives :

(a) Consistent positioning of the workpiece in relation to the tool despite all variables

(b) Holding the desired position of the workpiece against tool forces

(c) Restricting deflection of the workpiece due to tool and holding forces or the weakness of the workpiece

(d) Achievement of the above without damaging existing machined surfaces.

2.1.1 Concept of location

When a workpiece is placed on a fixture, it is the function of the location system to provide the necessary constraint so that the workpiece is correctly located.

Figure 2 illustrates a body that is free in space. A body in this condition has six degrees of freedom; three of these are freedoms of translation, and three are freedoms of rotation. The location system must, in conjunction with the clamping system, completely constrain the workpiece, removing all six of its degrees of freedom.

To study the complete location of a workpiece within a fixture, consider a rectangular block that is free in space. Place three points, not in a straight line, on the bottom surface. These three points stop motion in the vertical direction and also prevent rotation around the longitudinal and the crosswise axis as shown in figure 2. In other words, they have deprived the block of three degrees of freedom. The

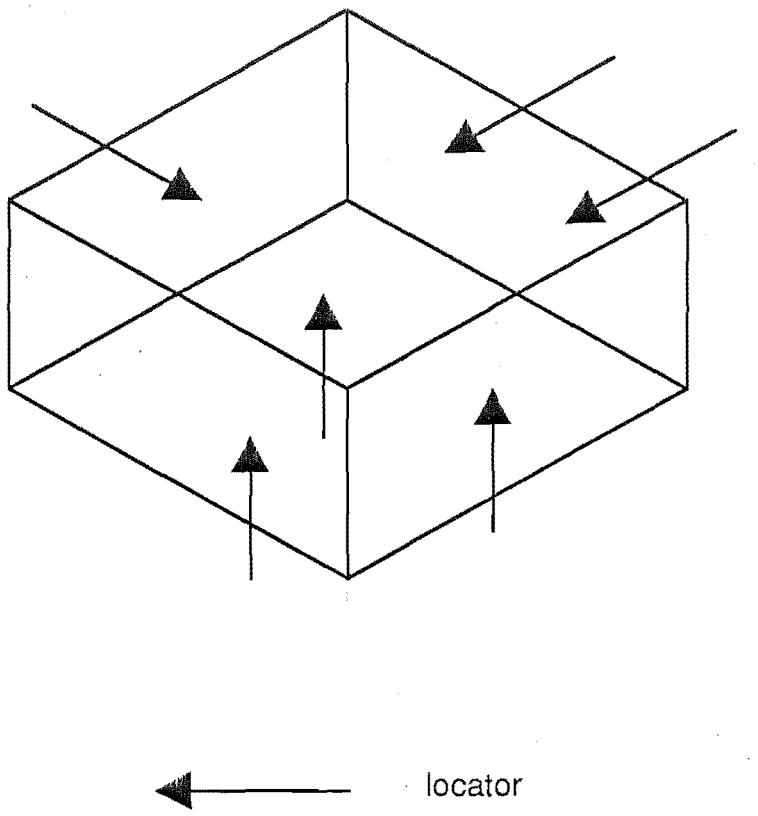


Figure 2 "3-2-1" Location system

block can still slide in two directions in the plane defined by the three points, and can rotate around a vertical axis. Place another two locating points, not in the same vertical line, on one of the vertical sides. These locators stop linear motion of the block against the locators and also rotation about the vertical axis (figure 2), thus depriving the block of two degrees of freedom. Finally apply one locating point against a side orthogonal to the first two sides; this eliminates the sixth degree of freedom and the block is now fully located.

The above method of locating a workpiece in a fixture is called the "3-2-1" principle. There are many other ways to eliminate the six degree of freedom, each of which is equivalent to the 3-2-1 locating system.

2.1.2 Support and clamping

In addition to locators, supports are sometimes needed to reduce deflections. Part of the workpiece may lack adequate rigidity to withstand the cutting forces without deflecting, or the workpiece may simply be too large to be adequately supported by the locators. They may then deflect under the tool and clamping forces, or under even their own weight. This causes machining inaccuracies and possibly elastic deformation of the workpiece. As already mentioned, supports are used to reduce deflections. It is important to note that supports must not interfere with the locating of the workpiece which has already been established. The supports are, therefore, often made adjustable

and brought against the workpiece without significant pressure and without producing any geometric redundancy.

The clamping system must hold the workpiece firmly against the locators so that it cannot be moved by cutting forces. One important rule is that the clamping force must be applied as directly as possible, and without causing any elastic deformation of the workpiece. In the case of thin walled workpiece, it may be necessary to allow for additional clamps, so that each clamping pressure is transmitted through solid material preferably to its location point.

2.2 ROLE OF FIXTURE DESIGN IN PROCESS PLANNING

This section outlines the process planning procedure and the role of fixture design in process planning. A flow chart of the process planning procedure is presented in Figure 3.

The task of process planning usually comprises of a series of steps. The first of which is to interpret the part drawing. It involves looking at the general description, the size and shape of the workpiece, the material specifications, the revision of the drawing and general notes in the drawing. Then according to the interpretation, the machining process is determined. Next, the machine centers which usually can carry out one or multiple machining processes have to be selected. The selection should consider availability, process capability, range of machining operations, production rate, etc. Then the operation sequence must be determined.

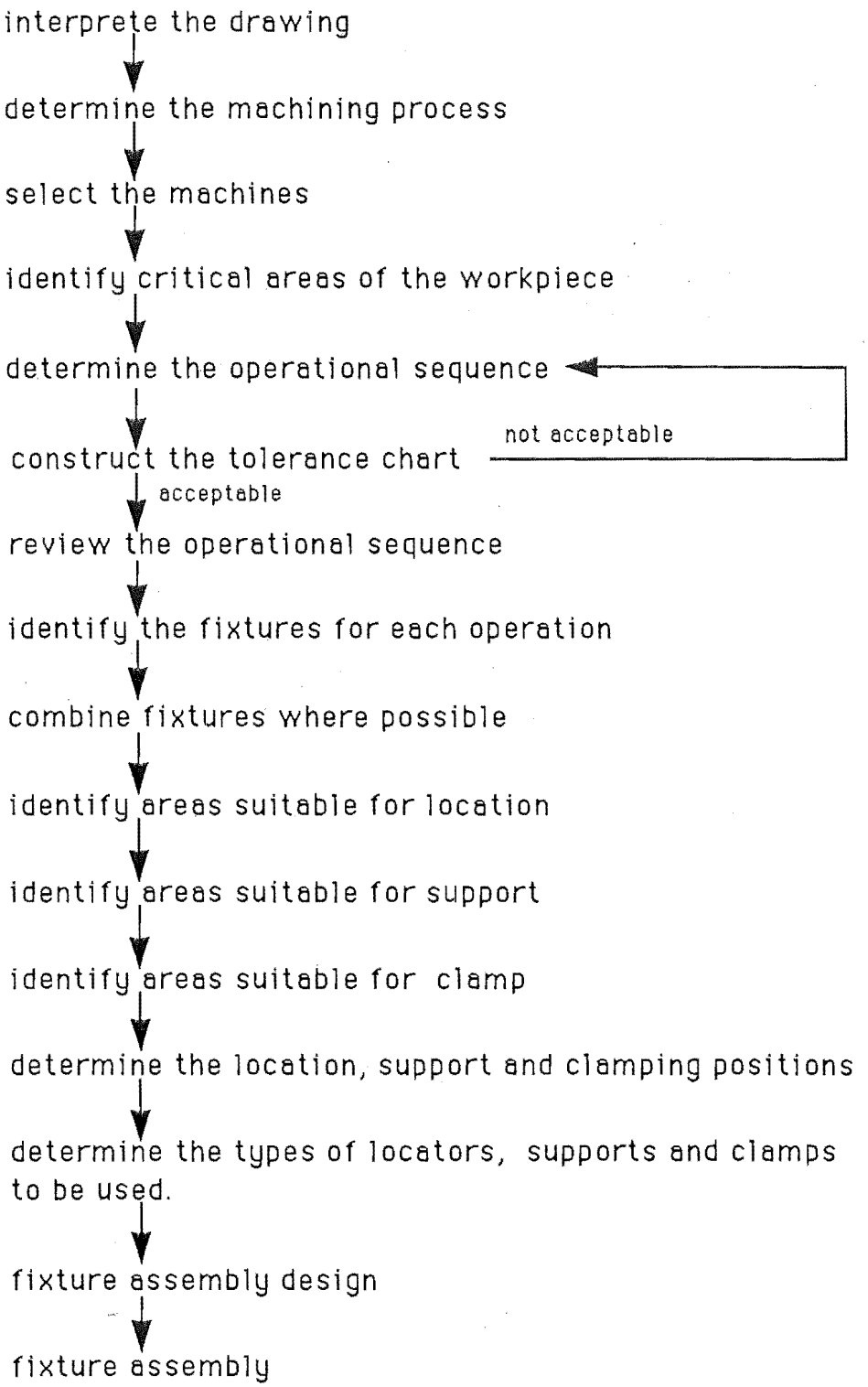


Figure 3 Process planning procedure

2.2.1 Determining the operation sequence

It is important in planning the sequence to identify all the critical areas of the workpiece. There can be two types of critical areas on a workpiece [7]. One type is called a process critical area. It is used to provide location areas for the workpiece in subsequent machining operations. The other is called a product critical area, and is identified by one or more demanding functional specifications on the part print. Often, a surface on the workpiece can qualify as both a product and process critical area. The critical surfaces identified, act as a constraint for determining the manufacturing sequence. One cannot adequately consider performing other operations without repeated reference to those critical areas.

The best sequence of manufacturing operation is determined by the degree of control which can be maintained throughout the process and the logical process order. Therefore, it is important in planning the sequence to get to the critical process areas first, to assure that the important dimensional relationships of the workpiece can be maintained throughout the balance of the process sequence. The position in the sequence of other manufacturing operations are dictated by rules and logical process order.

Examples of rules:

(1) Process critical areas must be accomplished early in order to maintain control over the dimensional relationships of the workpiece during subsequent operations - an important rule.

(2) Delicate surfaces should be created late in the operational sequence in order to protect them from damage or accident in other operations.

(3) Areas identified by close dimensional tolerances are accomplished as early in the operational sequence as possible for economic reasons. Close tolerances are more difficult and costly to produce, and are more likely to cause the part to be scrapped. Therefore it should be scrapped before other operations can be performed on it.

Examples of logical process order:

(1) The sequence of drilling followed by reaming

(2) The sequence of turning followed by threading

(3) The sequence of milling followed by grinding

2.2.2 Determining areas used for fixturing

The next step is to review the manufacturing sequence, and to identify the surfaces to be machined, and the surfaces with which the part can be located, supported, and held for each operation. Fixtures should be combined, where possible, by grouping those with common location system, to reduce cost.

Areas suitable for location

The process critical area is used to provide location areas for the workpiece in subsequent machining operations. Therefore it is important in planning the sequence to get to the critical process areas first, in order to assure that the important dimensional relationships of

the workpiece can be maintained throughout the balance of the process sequence.

Areas suitable for support.

As discussed in section 2.1.2, part of the workpiece may lack adequate rigidity to withstand the cutting forces without deflecting. Forces created by the cutting action deflect the unsupported workpiece causing the tool to chatter. The result is poor surface finish and loss of dimensional control.

The addition of supporting elements to the workpiece should take place after the locating surfaces are established. Some important guides for selecting supporting areas are as follows:

1 Select supporting areas on the workpiece where maximum deflection is likely to occur.

2 Select areas which will not interfere with the locating areas, and those areas that are to be machined.

Areas suitable for clamping.

The clamping system must hold the workpiece firmly against the locators so that it cannot be moved by cutting forces. Because the order for establishing the processing areas leaves holding until last, the choice of suitable surfaces may be limited. The ideal surfaces for holding or clamping are those directly opposite the locating points. Because those surfaces are the ones which are also most likely to be the ones requiring the machining, an equivalent resultant holding force on other surfaces must be obtained.

Some fundamental guides for selecting surfaces for holding the workpiece are as follows:

1 Avoid using areas that are to be machined for clamping, unless the machining does not include the entire surface.

2 When possible, choose surfaces for clamping opposite those for location.

3 When the surface opposite the locators must be machined, choose alternate surfaces in such a way that an equivalent resultant clamping force can be established.

4 Choose surfaces which will not cause the clamping action to distort the workpiece.

5 Choose surface areas large enough to distribute the clamping forces, because holding forces concentrated in small areas often damage the workpiece.

6 When possible, avoid using previously machined areas for clamping when there is the possibility that they might become damaged when applying the holding force.

2.2.3 Fixture design and manufacture

Several different adaptable fixtures are usually available to handle components from the various part families. However, if no suitable fixture already exists, design and manufacture of a new fixture is necessary. With the use of a modular fixturing system, the fixture can be assembled within hours of the component being available, therefore reducing the fixture design and manufacture lead time.

CHAPTER 3

COMPUTER AIDED DESIGN OF MODULAR FIXTURE

The review of the literature pertaining to particular aspects of the computer aided design of fixtures is included in each appropriate subsection in this chapter.

For efficient production, fixture design should not be viewed in isolation but as an integral part of process planning. For the purpose of this study, however, it is assumed that the process planning has already been completed when the fixture design and manufacture starts.

For modular fixtures, the fixture assembly design and assembly may be divided into three stages as shown in Figure 4. Modular fixtures, which has been introduced in section 1.1.3, generally refers to fixtures which are assembled from a set of precise modular elements.

3.1 STAGE 1 : FIXTURE DESIGN

The first stage, the fixture design, is closely linked to the design of the process plan. Surfaces identified as reference surfaces in the tolerance chart [71] are selected as location surfaces and the various fixture elements, locators, clamps, and supports are chosen and positioned to satisfy the rules of workpiece control as outlined in Chapter 2. Since this problem lacks a systematic structured procedure, an expert system may be suitable for this application. An

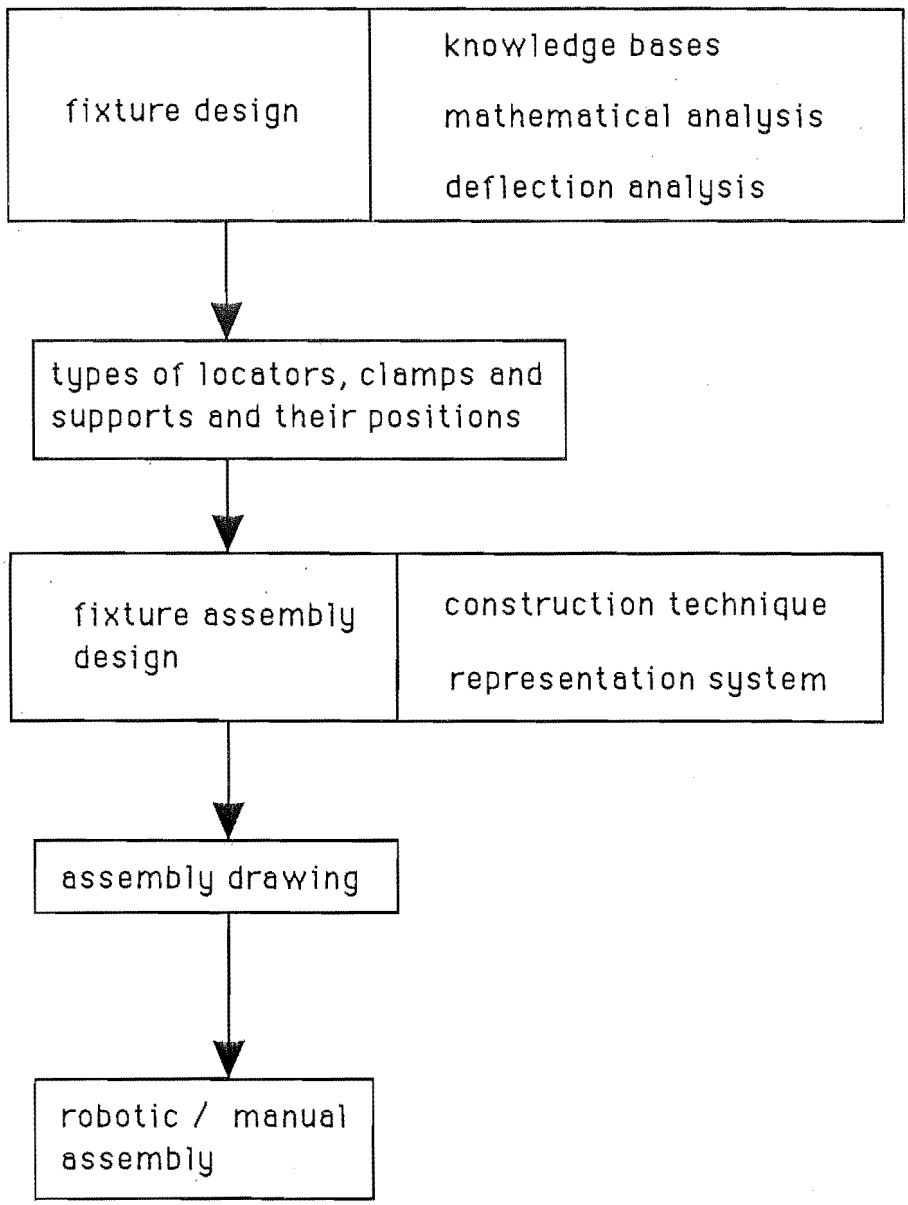


Figure 4 Modular fixture design and assembly

expert system is able to capture the expertise and knowledge as displayed by human experts.

An expert system is defined by Clive L. Dym as a computer program that performs a task normally done by an expert or a consultant, in so doing it uses captured, heuristic knowledge [45]. An expert system has a knowledge base, an inference engine, and an input/output mechanism. These components of the expert system has been described by John E. Biegel [50] and are summarised in the following section.

The knowledge base contains domain-specific knowledge. AI researchers often divide knowledge into "surface" and "deep" component [44,45]. The term deep knowledge is used to refer to reasoning from basic principles. Surface knowledge is that heuristic, experiential knowledge that comes from having successfully solved a lot of problems.

The inference engine [50] does the reasoning for the system. The reasoning frequently is done by "production rules". Production rules are of the form " if...then". For example, *if* it looks like rain *then* take an umbrella. This is an example of a forward-chaining rule. The inference engine uses both forward and backward chaining procedure. When a chess player looks at a problem, he reasons forward from the current state of affairs toward a solution. This is called forward chaining. When a medical doctor faces a problem to be solved, he sees symptoms and tries to reason backward to find probable causes. This is called backward chaining.

The input/output mechanism [50] communicates with the outside world. The input/output mechanism allows the user (includes expert) to communicate with the system. It requests and accepts input, puts the input in the proper form and sends it to the inference engine or the knowledge base. It outputs queries or solutions to the user.

Various researcher, discussed later, have applied this approach to select the required locators, clamps, and supports and to determine their positions. To develop a fixture design expert system, the fixturing concepts and rules form an important part of the knowledge base (Refer to section 2.1 and 2.2.2). These fundamental concepts and rules (deep knowledge) can be gathered and consolidated from various fixture design textbooks (for example, Eary and Johnson[7], Hendrikson [8], Donaldson, Lecain and Goold [16], Boyes[17], Jones [11], Wilson [13], Sharma [9], Sedlik [10], Ryder [18] and Kempster [14,15]). The knowledge base of this expert system would also have heuristic, experiential knowledge from experts in fixture design (surface knowledge). Some aspect of the design may be subjective, however, the input/output mechanism allow the expert system to interact with the user.

In addition to the qualitative requirements outlined in section 2.1 and 2.2.2, quantitative procedures are also involved in the design of fixtures. For instance, although it is known from the 3-2-1 rule that six locators are generally required to locate a component, their exact co-ordinates are not given by that rule, and therefore have to be determined. Similarly, the positions of the clamps also have to be decided.

The external forces, the clamping forces and the friction forces can be mathematically expressed by a resultant force vector (R) and a resultant moment vector (M) about the center of mass of the workpiece. Figure 5 shows a free body diagram of a workpiece supported by the fixture. In order for the workpiece to remain in equilibrium, this resultant force and resultant moment imposed on the part, have to be absorbed by a system of locators. Since a general unconstrained spatial motion of a rigid body has six degrees of freedom, the fixture must impose constraints to ensure that the workpiece has zero degree of freedom and hence, does not move.

The mathematical equation for analysing the position of the locators and clamps, and determining their optimum placement, is expressed by the following equations for static equilibrium:

equation [1] : sum of forces = 0

equation [2] : sum of moments = 0

The above equations have to be solved to determine the location and the clamping points. The solution of the equation may be obtained using standard numerical methods such as Gaussian elimination [26].

Since the above mathematical analysis can only determine the location and clamping points, the deflection analysis is required to determine the support points. As discussed earlier, supports are only required for those workpieces with some weak points. The deflection analysis could be used to determine the point where maximum deflection occurs. A support would be placed at the point of maximum deflection.

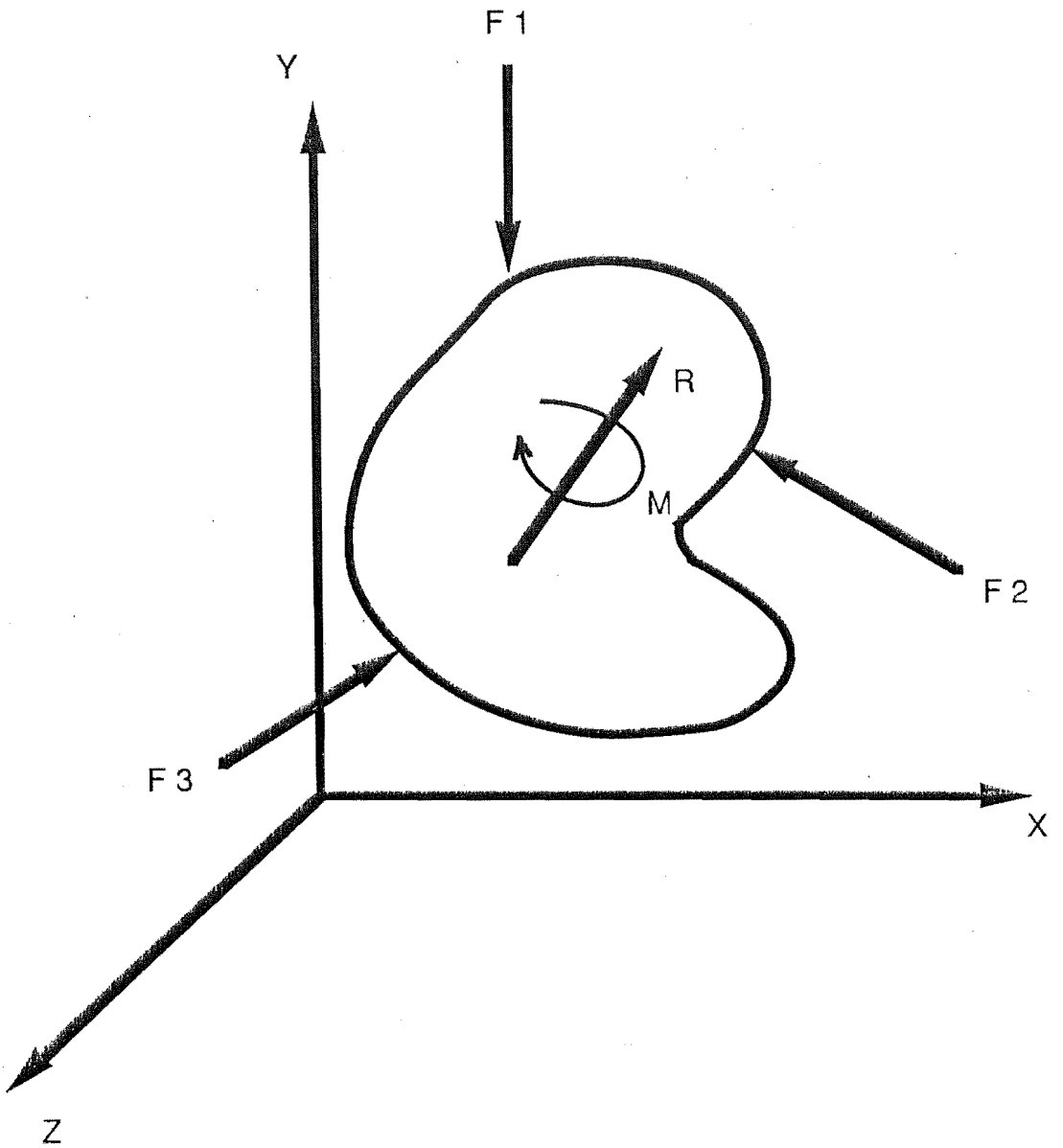


Figure 5 Free body diagram of a workpiece

In recent years, computers have played an increasingly important role this aspect of fixture design. Pham, Nategh and Sam Lazaro [30] have developed a jigs and fixture knowledge-based system. The program is implemented in an expert system shell. It incorporates design principles, as described above, as well as numerical procedures. The interactive knowledge-based program is used to assist designers of jigs and fixtures to determine the required locating, clamping and support positions.

Gandhi and Thompson [26] proposed to incorporate the knowledge of experience personnel in jig and fixture design, in addition to the design principles and numerical procedures. An automated assembly design process, discussed in the next section, is also proposed.

Nee, Bhattacharyya and Poo [28,31] investigated using a knowledge-based approach to select the appropriate fixture elements. These elements are then assembled interactively to form the final fixture.

3.2 STAGE 2 : MODULAR FIXTURE ASSEMBLY DESIGN

In the second stage, modular fixture assembly design, the body of the fixture which holds the locators, clamps and supports in position on the baseplate is designed. The body of the fixture is constructed from a set of blocks mounted rigidly onto a baseplate.

Locators, clamps, and supports are elements of the fixture in direct contact with the workpiece. The body of the fixture holds these

elements together rigidly and accurately in position to perform their functions.

With the present state of art, all elements of the modular fixturing set are stored on a computer data file. The tool designer brings the workpiece on to the screen, selects the elements required from the data and then visually builds up the fixture around the component. On completion of the design, the tool designer instructs the CAD plotter to produce a drawing of the fixture. The drawing is then given to the operator for assembly.

Markus, Markusz, Farkas and Filemon [25] have developed an expert system, written in Prolog, to design the modular fixture assembly. A tower design program is used to design and store a collection of towers. Each tower consists of a few blocks connecting a locator, clamp, or support to a slot on the baseplate. A few suitable towers are retrieved for each locator, clamp, or support. The fixture is then generated using a multilevel refinement scheme. A separate data structure is used to store shapes of the workpiece, machining envelope and the fixturing elements for the purpose of collision test.

Gandhi and Thompson [26] proposed using the geometrical features of the fixturing elements and the spatial relationships between these features to determine the assembly process of the fixture. The faces of the various components has geometrical features, such as face, edge, shaft, or a hole. The contact between these surfaces are governed by certain rules about spatial relationships, such as against, fit, coplanar.

3.3 STAGE 3 : MODULAR FIXTURE ASSEMBLY

The final stage is to assemble the fixture components in strict accordance with the fixture assembly design. Some progress has been made towards using robots for automatic assembly of the fixture [34,40]. This is made possible by using a limited set of fixture components specially designed for automatic assembly. It is unlikely that the complex set of components used in most commercial modular fixture systems could economically be assembled by robots.

3.4 APPROACH ADOPTED

The work described here provides a computer aided method of designing the fixture body, using a set of standard modular fixturing elements (stage 2). The work by Markus, Markusz, Farkas and Filemon [25], the pioneers in this field, is the only other significant contribution. They use expert system technique to design the modular fixture body from a huge library of towers. Unlike Markus, a deterministic approach has been adopted here to provide a systematic procedure. Instead of storing towers in the knowledge base, it stores the rules required to construct towers.

3.4.1 Brief description of the method

The inputs to the design system developed here are the part geometry and the position of the location, clamping, and support elements. The fixture body must provide the connection between the locator, the support, and the clamping elements of the fixture and the baseplate.

The design system automatically designs the fixture body, and provides assembly diagrams and listing of the components used.

The systematic design procedure consists of two major parts. The first part generates and positions a series of towers appropriate for holding the given locators from the set of modular elements, and the second part of the program mounts the towers onto the baseplate to form the required fixture.

If any intersection occurs during the tower construction, the program will consult a rule-base. The rule-base consist of "if.....then" rules governing such intersection. It may move, combine, eliminate and update 'towers'. The rulebase is also able to assimilate new rules to enhance its decision-making capability.

3.4.2 Spatial representation system

In order to describe the space occupied by the various components of the fixture/workpiece assembly, to establish continuity of connection and to check for intersection, a spatial representation technique was developed.

This representation permits the program to identify the type of intersection, to determine the relative position of the modular elements, and to the measure the distance between the modular elements during the design process. It also enables the program to move and combine the modular elements systematically to form a fixture.

3.5 SCOPE

Commercial modular fixturing systems are designed for manual assembly. For ease of manual assembly, the functions of some modular elements are duplicated. Furthermore, non standard features, which benefit manual assembly, are sometimes added to the modular elements. Clearly, the above factors will result in an inefficient construction system. Therefore, the basic set of elements used in this project was carefully selected and adapted from the CATIC modular fixturing system for the purposes of automated construction. All elements in this system are keyed together using high tensile steel 'T bolts' and mounted onto the baseplate.

Figures 6 and 7 give an impression of the fixturing system used.

To illustrate the design method and to ensure that the program is of manageable size and efficiency, the following specifications were used:

(1) The shapes of the workpiece and the tool should only be specified to the detail actually needed for fixturing, i.e. the geometric model may hide all irrelevant details.

(2) The fixture is to hold and to position a prismatic workpiece. It is to be built according to the 3-2-1 principle [7,8].

(3) Any locator, clamp or support may be used in the fixture provided it satisfies the following conditions:

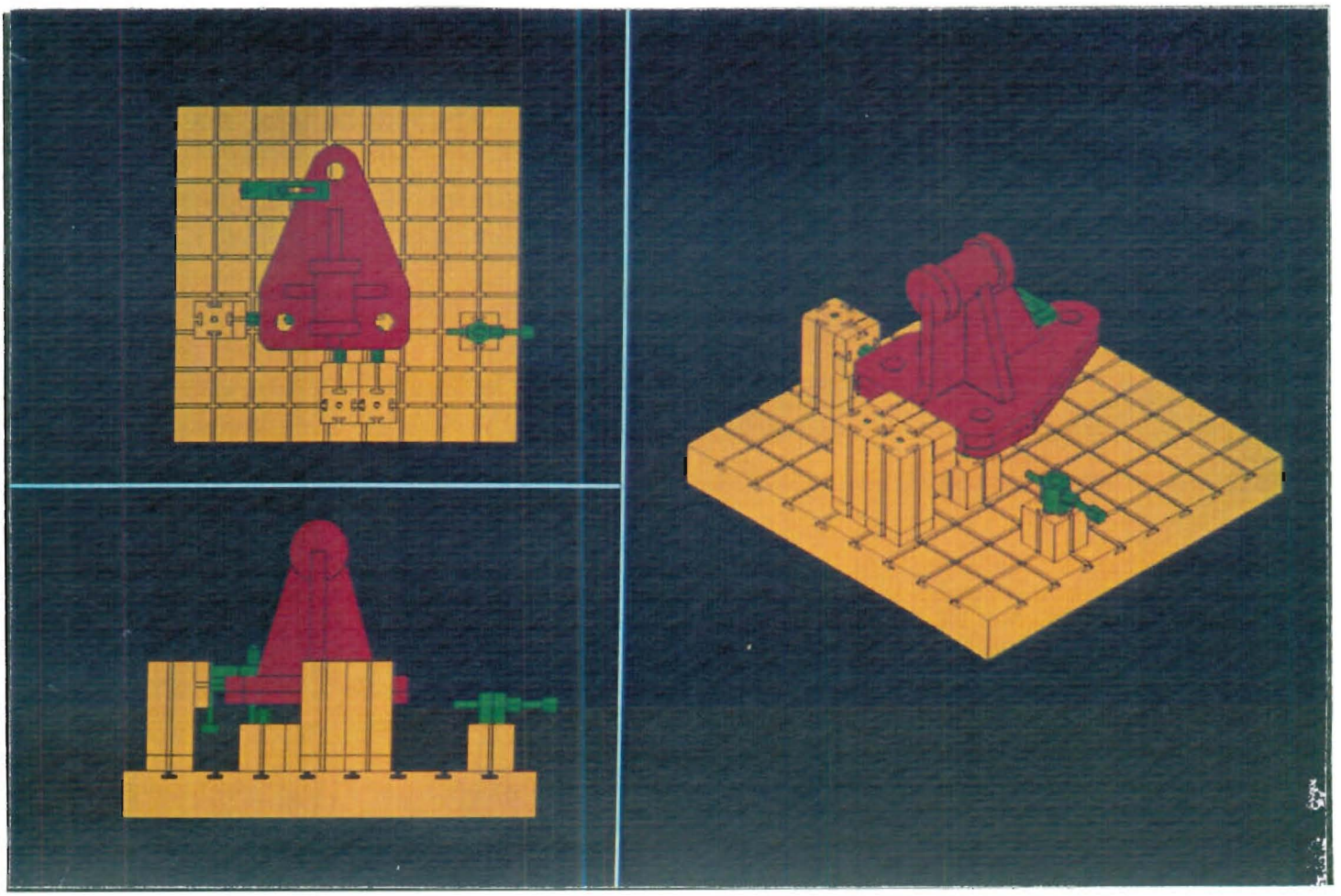


Figure 6 An example of the modular fixture used

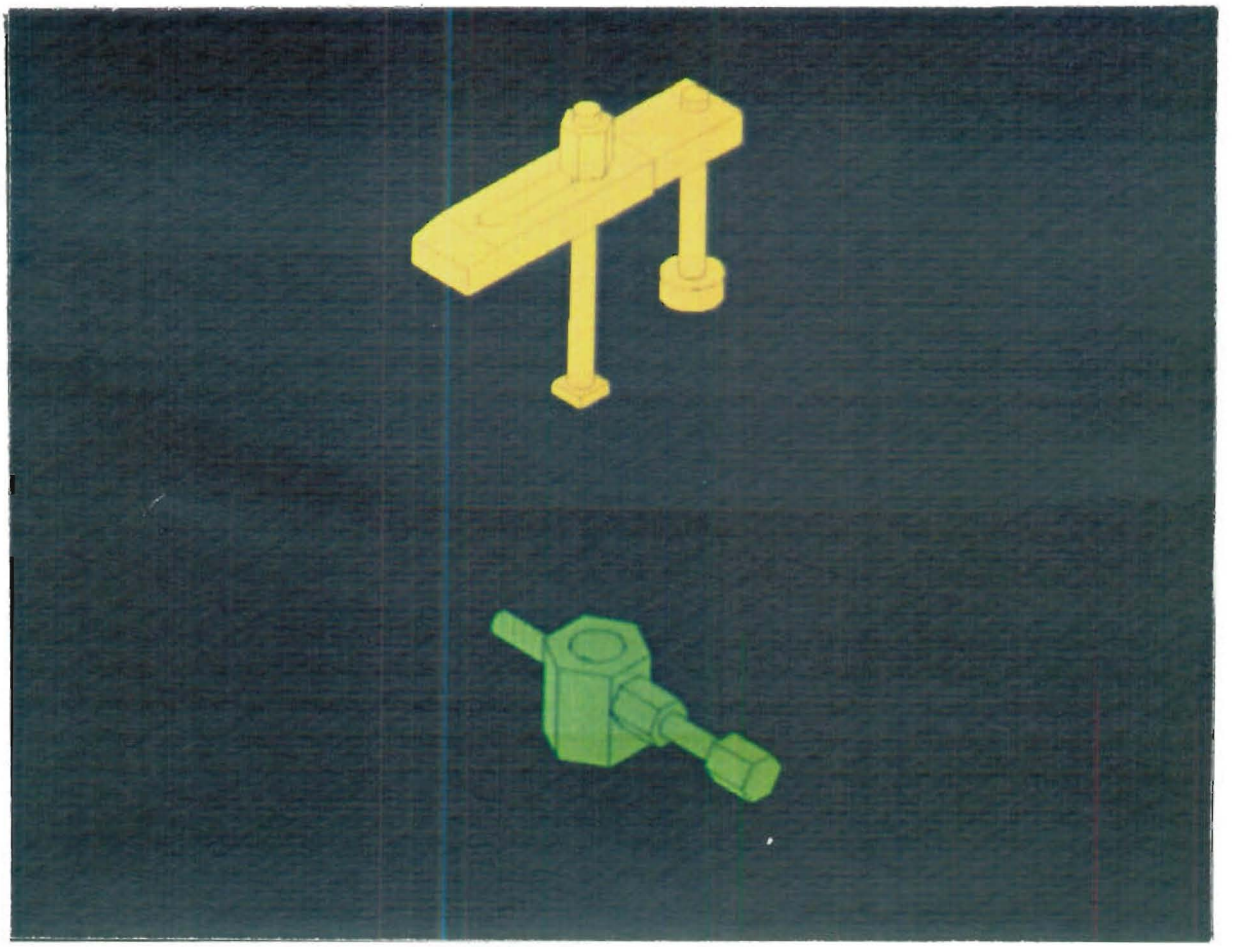


Figure 7 Types of clamping element

(a) It contacts the fixture body in a horizontal or a vertical plane as illustrated in Figure 8.

(b) The point of at which the locator, clamp, or support are held by the fixture body is determined.

(c) The cross section of the locator, clamp, or support, at the region of contact, should generally be the same as the modular blocks (for example 60mm X 60mm). An exception to this condition is the "button" locators, clamps, and supports.

(d) The locator, clamp, or support should have a hole that is compatible with the modular blocks so that it can be fastened to the fixture body via a 't-bolt'.

Some examples of the locators, supports and clamps are : the "Vee-block" supports, the "button" locators, the "button" clamps and the "tower" clamps. The "button" clamps and the "tower" clamps are illustrated in figure 7.

(4) The fixtures should be built in advantageous positions only, that is, the slots of the baseplate should be exploited as much as possible.

(5) Any element of the fixture must not have collisions with a solid. The solid includes the workpiece, the machining envelope, the locators, clamps, and supports and other elements of the fixture.

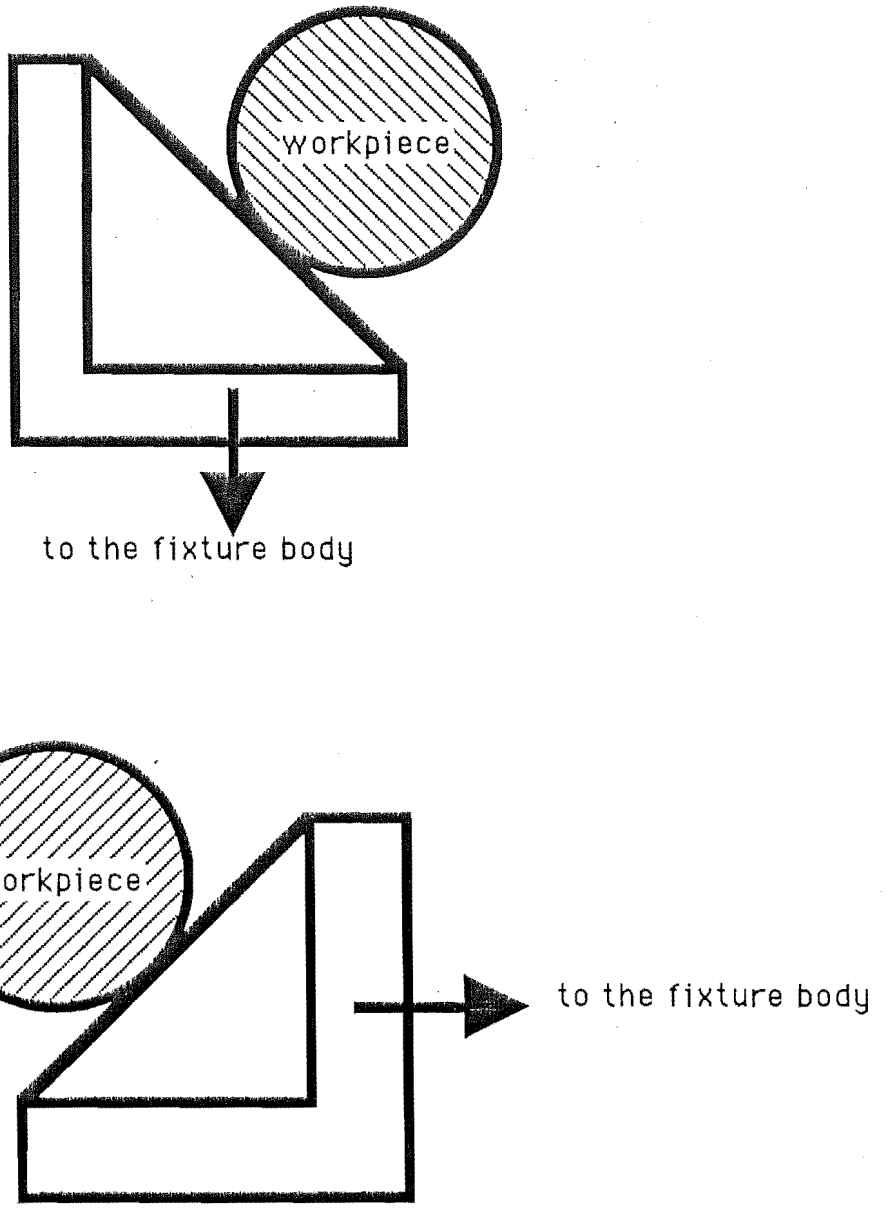


Figure 8 **Examples of vertical and horizontal face locators contacting the workpiece at an oblique angle**

CHAPTER 4

FUNDAMENTAL CONCEPTS

Having studied the available modular fixturing systems, the design procedure for the modular fixture assembly has been rationalised and is described systematically in the next few chapters. This chapter introduces the fundamental concepts of the method.

4.1 MODULAR FIXTURES

The modular fixturing elements may be divided into 3 groups:

(a) Locators, clamps and support

(b) The baseplate

(c) Modular blocks and shims

4.1.1 Locators, clamps, and supports

The locators, clamps, and supports fall into several different categories. These are:

(1) Horizontal face locators, clamps, and supports.
These attach to the body of fixture at a horizontal plane and are generally subject to vertical forces only.

(2) Vertical face locators, clamps, and supports.
These attach to the body of the fixture at a vertical plane and are generally subject to horizontal forces.

(3) Thrust locators.

These are a special case of a vertical face locator. They resist the machining or clamping loads placed on the workpiece. For this reason, they must be attached to the baseplate in as direct a way as possible.

4.1.2 Baseplate

The baseplate is a plate which provides an accurate mounting surface for the complete fixture. In the case of the CATIC system, the baseplate has numerous T-slots machined at right angles across its mounting face. They are machined exactly perpendicular and parallel to each other. A typical example of a baseplate is shown in figure 9.

4.1.3 Modular blocks and shims

The modular blocks and shims are then used to form the fixture body which provides a connection between the baseplate and the locators, clamps, and supports. Assuming that strains are small enough, only the rigid shapes of the elements are dealt with.

The modular blocks used in this project are of cross section 60mm square and of various heights. Each block has a vertical hole passing through the 'top' and the 'bottom' faces. The other four vertical faces are machined with slots parallel to the axis of the hole. Figure 10 shows a schematic drawing of a typical modular block. For the purpose of connectivity, the topology of the blocks is important.

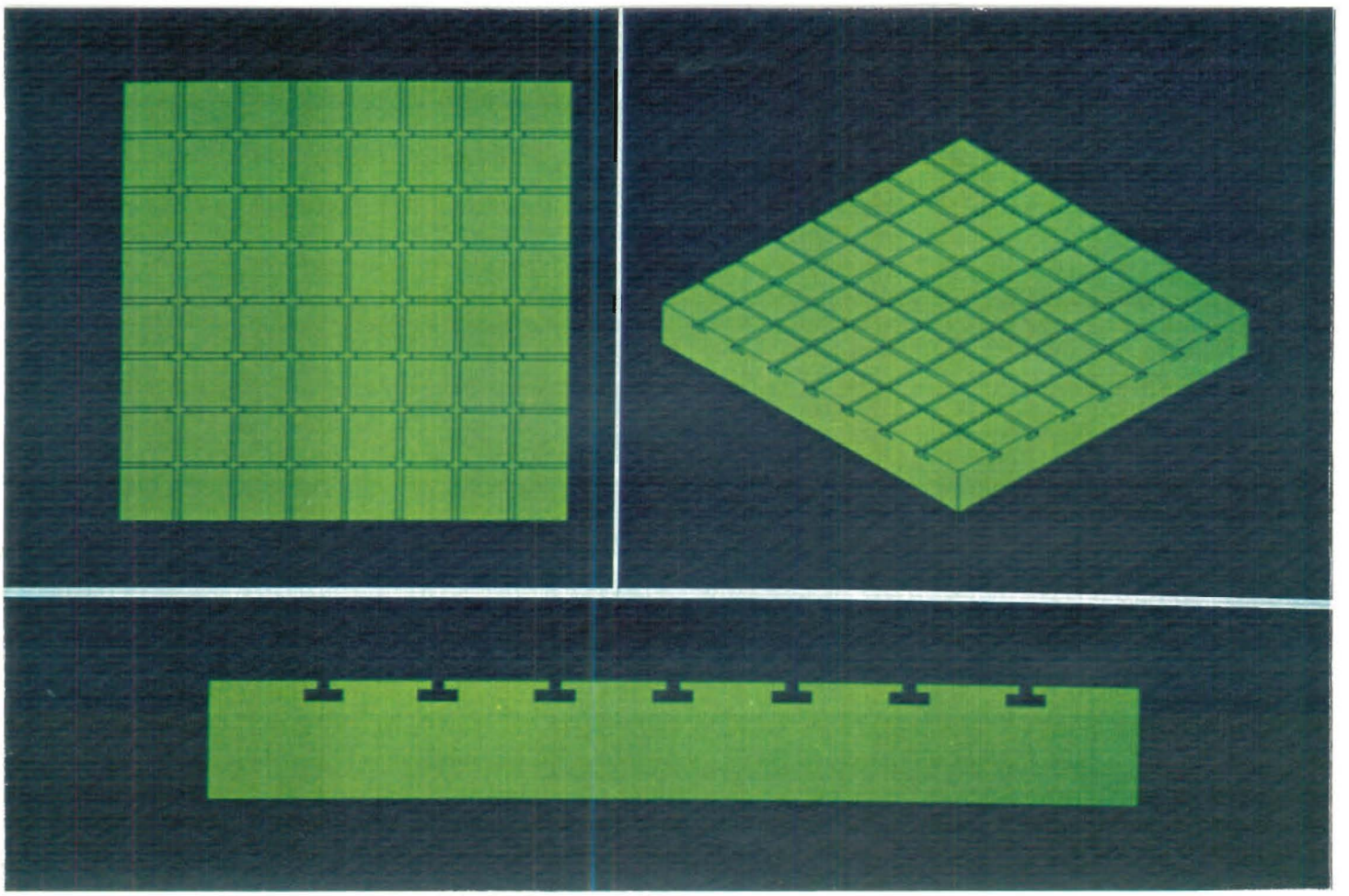


Figure 9 A typical example of a baseplate

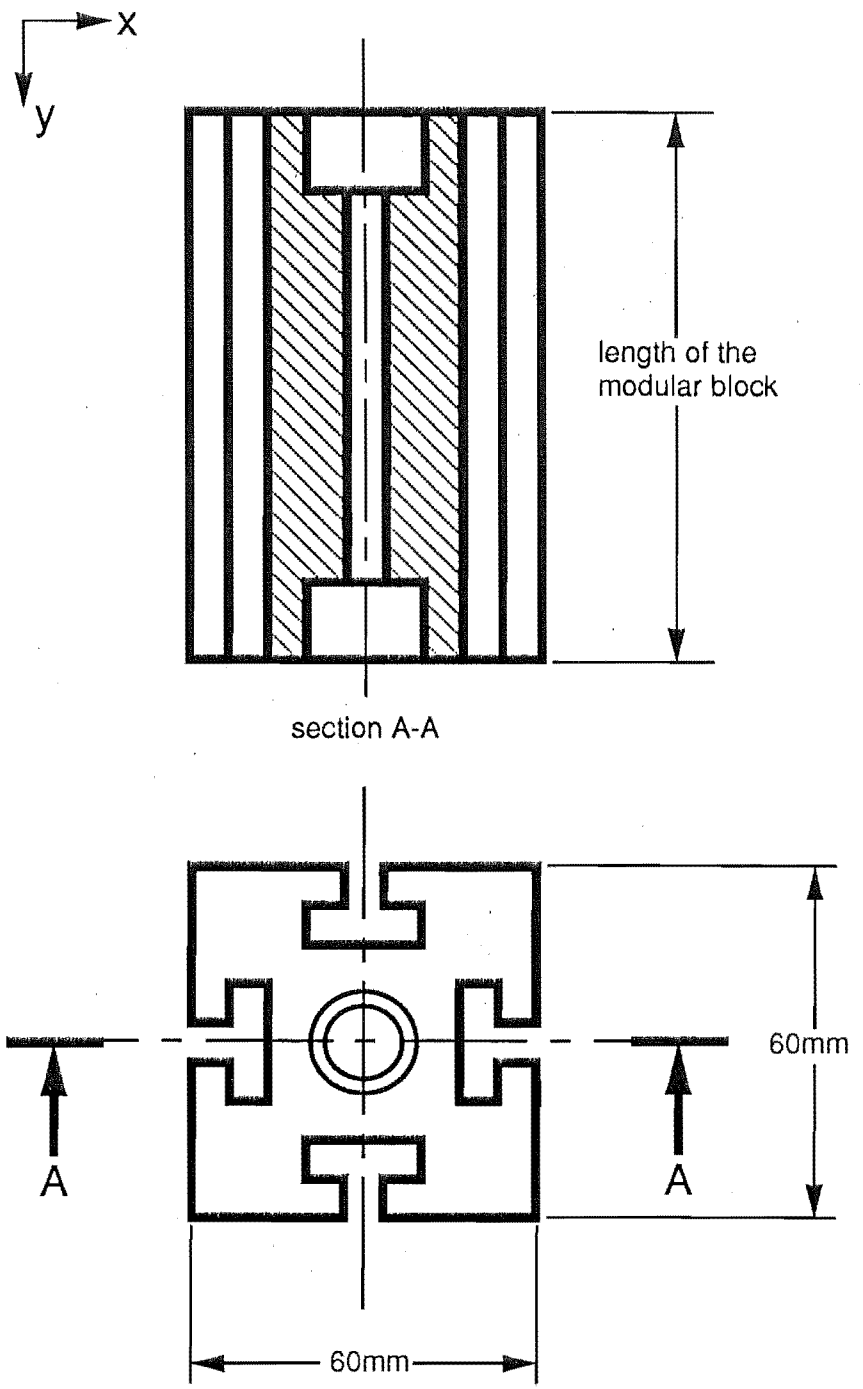


Figure 10 A typical modular block

Shims have the same cross section as modular blocks, but have a thickness of less than 10 mm. Normally the combination of blocks and shims would give the required dimension of a block.

4.1.4 Concept of slot and hole

Different configurations are used in different modular fixturing systems but the basic concept of the slot and holes is the same. For example, a hole can be connected to a slot but a slot cannot be connected to another slot. Constraints of this type must be incorporated in any design system for the fixture body.

The slot on the modular block is restrictive in terms of connectivity because it can only be connected to a hole; however it does allow sliding (continuous movement) along the slot.

In contrast to the slot, the hole on the modular block is flexible in terms of connectivity because almost any face can be connected to a hole; however it is inflexible in terms of relative movements between modular elements.

The slots on a baseplate are machined exactly perpendicular and parallel to each other, therefore they allow continuous movement in one dimension and discrete movement in the other.

4.2 FUNDAMENTAL STRUCTURES

4.2.1 Tower and mounting block concept

The three-block structure , as shown in Figure 11, is the most general structure. Each block may consists of a few modular blocks and shims held together by a single high tensile bolt. The combination of modular blocks and shims gives the required length of a block. The three blocks in orthogonal arrangement are able to reach any point on the surface of the workpiece from a slot of the baseplate. The x-y-z dimension is achieved by adjusting the length of the blocks in two directions and sliding the structure along the 't-slot' of the baseplate in the third direction. The connectivity constraint of the structure, discussed in section 4.2.2, is satisfied by the orthogonal arrangement.

The three-block structure consists of a tower and a mounting block. The block that is attached to the baseplate is the mounting block, while the other two blocks form the tower as shown in Figure 11. This tower may hold either a vertical face or a horizontal face locator, clamp, or support. The tower that holds a horizontal face locator, clamp, or support is called a horizontal face general tower (tower B), while the tower that holds a vertical face locator, clamp, or support is called a vertical face general tower (tower C).

Besides the general three-blocks configuration, in some cases two-blocks structure or even one-block structure may be used.

The two-block structure is made up of a special tower and a mounting block as shown in Figure 12. The special tower is called the horizontal

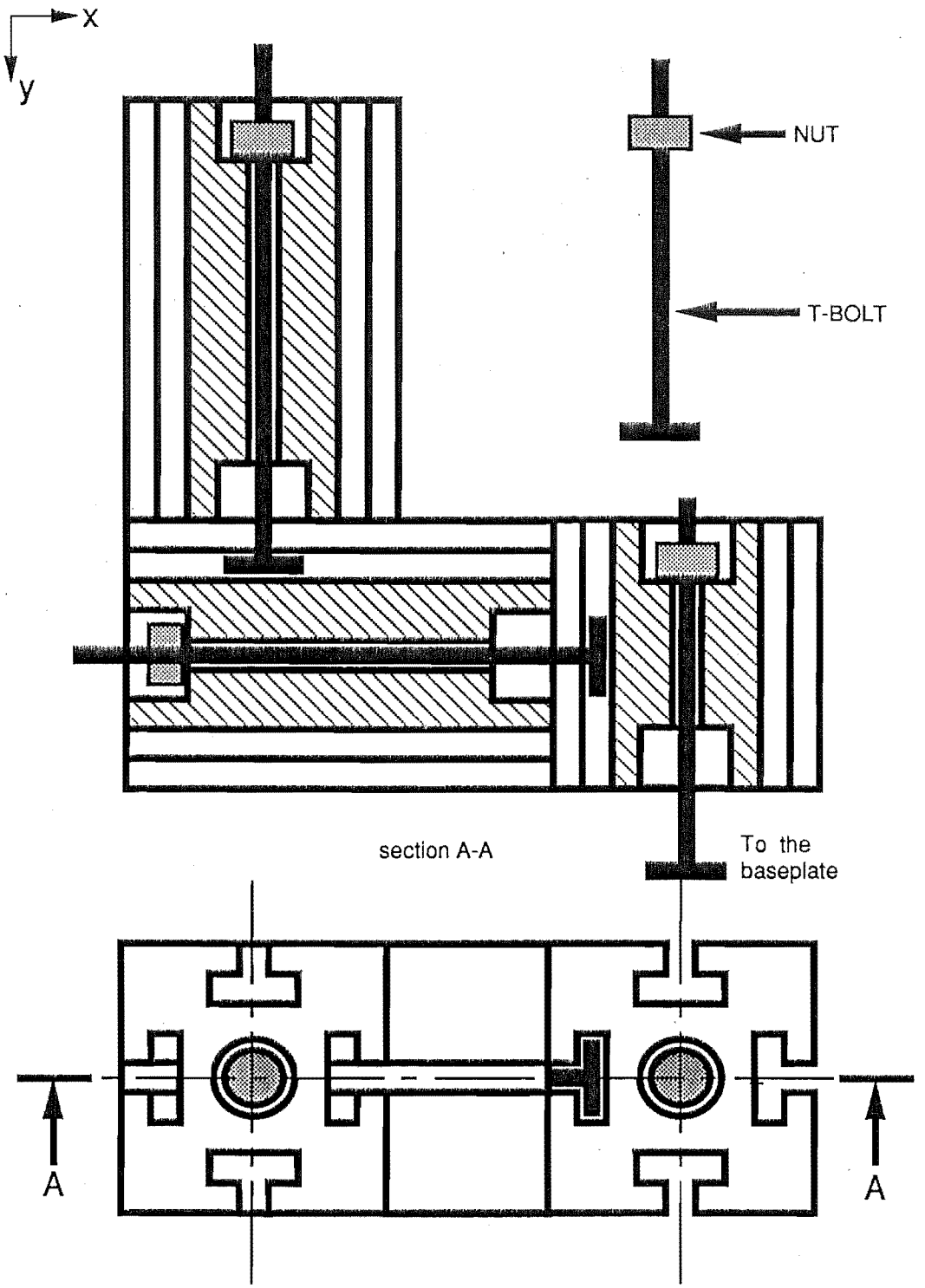


Figure 11 The three-block structure

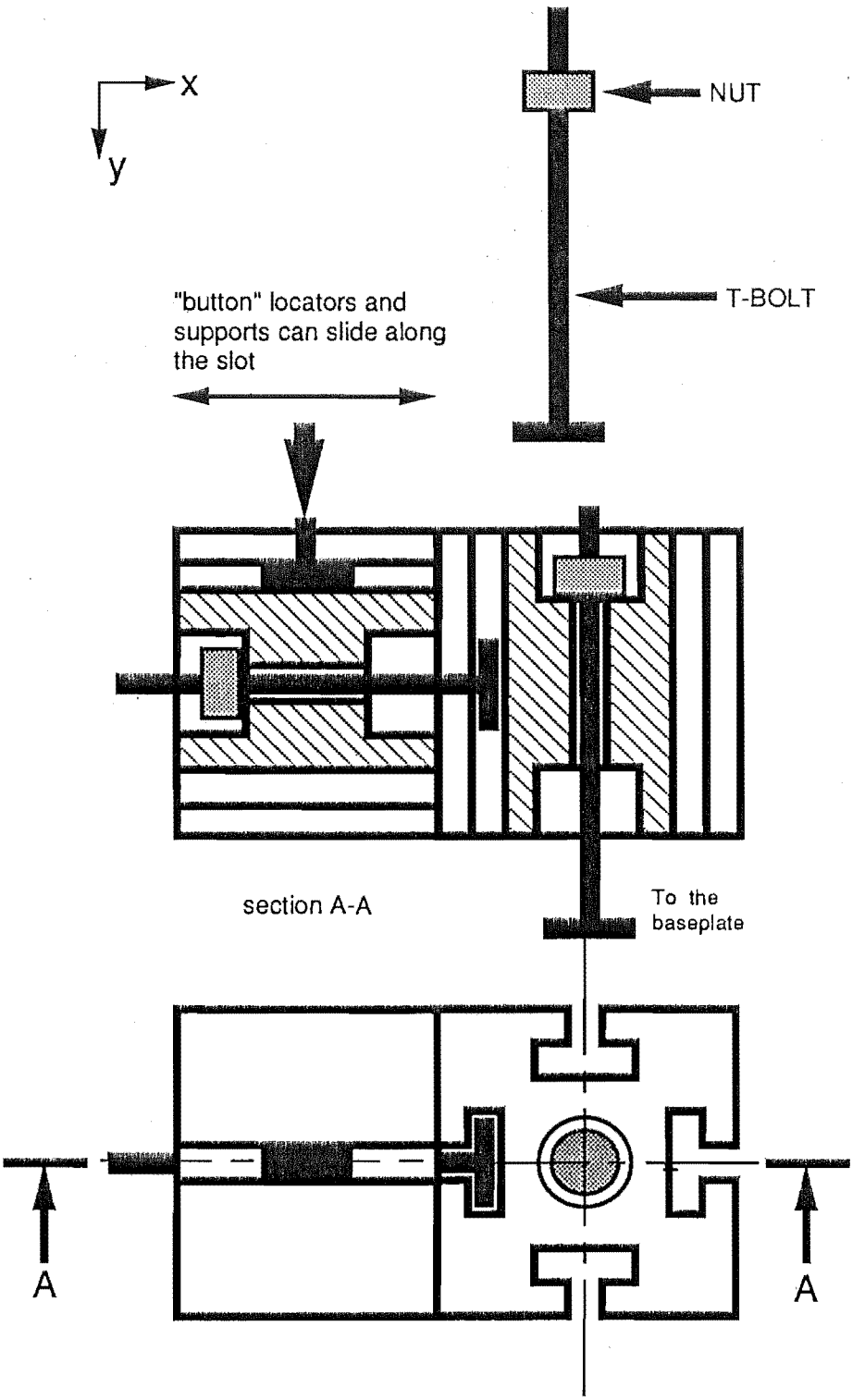


Figure 12 The two-block structure

face special tower (tower A). The horizontal face special tower occupies the least space. However the special tower can only be used if a horizontal face locator, clamp, or support is positioned at a certain height above the baseplate. This height above the baseplate is equal to the width of a block and is called the Design level.

In the one-block structure, the single block serve as both the tower and the mounting block. The advantages of using a one-block structure are :

(a) It does not require a separate mounting block.

(b) It is connected directly onto the baseplate allowing maximum rigidity.

However only towers that are aligned with a slot of the baseplate can use the one-block configuration. The one-block structure is usually a thrust tower or a direct tower. A thrust tower (tower D) holds a thrust locator and is always fastened directly to a slot of the baseplate. A direct tower is any other tower that is mounted onto a slot of the baseplate directly.

4.2.2 "Bridging-blocks"

"Bridging-blocks" are those modular blocks and shims used to connect the tower to the vertical face locator, clamp, or support. A combination of modular blocks and shims is used to bridge the gap between the tower and the locator, clamp, or support whenever the tower is moved away from the locator during the design process.

4.3 TOWER INTERSECTION AND RULE-BASE

There are four types of towers, which have been discussed in section 4.2.3, namely; the vertical face general tower (tower C), the horizontal face general tower (tower B), the thrust tower (tower D), and the horizontal face special tower (tower A).

The towers sometimes do intersect each other, and the towers may combined to form a combination tower or one tower may move until it is clear of the other tower. This construction process requires the detection and identification of the intersection, and the consultation of the rule base for the actions to be taken. Details of the rule-base be discuss in detail in the next chapter.

A spatial representation system is used to represent the space occupied by the workpiece, the machining envelope and the various elements of the fixture. It is also able to detect intersection, and identify the individual components involved in the intersection. The detection and identification is made possible with the use of a special number systems which will be discussed in Chapter 8.

4.4 TOWER MOUNTING

When all the towers are constructed, the final step is to mount these towers onto the baseplate. The tower can either be mounted directly or indirectly onto the baseplate.

Direct mounting occurs when the tower is aligned with a slot of the baseplate. Such a tower is fastened directly onto the baseplate with a single bolt, and is called a direct tower as described in Section 4.2.3.

Indirect mounting occurs when the tower is not aligned with a slot of the baseplate. Such a tower has to be connected to the baseplate either through another tower or through a mounting block.

Details of the tower mounting process is described in Chapter 7.

CHAPTER 5

OVERALL STRUCTURE

5.1 MODULAR FIXTURE DESIGN AND MANUFACTURE

The modular fixture design and manufacture process may be divided into three stages, as shown in Figure 4. During fixture design, the required locators, clamps, and supports are selected and their positions determined. The fixture assembly design section then builds the body of the fixture to hold the locators, clamps, and supports in position on the baseplate. The output of this section is an assembly drawing of the fixture. With the assembly drawing, the fixture is then assembled manually. The assembly of the fixture, given the assembly drawing, can be done within a few hours [52].

5.2 MODULAR FIXTURE ASSEMBLY DESIGN

The objective of the fixture assembly design procedure is to select combinations of blocks and shims to make the connection between the reference points on the locators, clamps, and supports and the baseplate, in such a way that the locators, clamps, and supports can fulfill their functions. It is important to note that the fixture body is designed using as reference points, the points at which the locators, clamps, and supports are held by the fixture body, and not the points where these elements actually contact the workpiece.

The fixture body therefore provides the connection between these elements of the fixture and the baseplate. Because of the immense

number of combinations of the elements, and the connectivity constraint of the various components of the assembly, it is inefficient to built the fixture element by element from scratch. It is observed that all fixtures have some common fundamental structures (section 4.2). Therefore it is possible to study the relationships between elements (holes/slots), and extract some basic combinations of the elements which are commonly used.

Considering the above factors, the fixture assembly design problem is divided into two main stages:

(1) The generation of fundamental towers, and the combining of these towers as required.

(2) The mounting of the towers onto the baseplate.

The development of the fixture assembly design program is a formidable task as the resulting structure must take into account:

(1) the connectivity between modular blocks.

(2) the limited number of 'T-slots' on the baseplate for fastening the primary blocks.

(3) the volume available for placement of the elements of the fixture body without interfering with space already occupied by the workpiece, or the machining envelope, or other part of the fixture body.

(4) structural stiffness requirements of the fixture body.

A suite of programs has been developed which completely automates the design procedure for the fixture body, and has been implemented in Turbo Pascal 4.0. The two main programs are the tower generation and the tower mounting programs. The main programs are supported by rulebases, utility programs, and subprograms. The overall structure of the program is outlined in the following sections and is represented schematically in Figure 13.

The automated design procedure requires a spatial representation system, that could detect and identify objects and its intersections automatically. The representation system must also be able to manipulate objects, and to interpret rules relating to the manipulation of the objects during the fixture design process. A matrix-spatial representation technique, which satisfies the above requirements, has been developed, and is described in chapter 8.

5.2.1 The input

The drawing of the input comprises the workpiece, the machining envelope, and the type and placement points of the locators, clamps, and supports. It is assumed that these have all been determined prior to embarking upon this stage of the design. The positions of the locators, clamps, and supports are specified in a frame of reference, which is fixed relative to the workpiece and the machining envelope.

The drawing of the input is converted into a matrix-spatial representation. Because of the memory constraint the conversion process excludes any irrelevant geometrical details. An example of the input and its corresponding matrix spatial representation is shown in

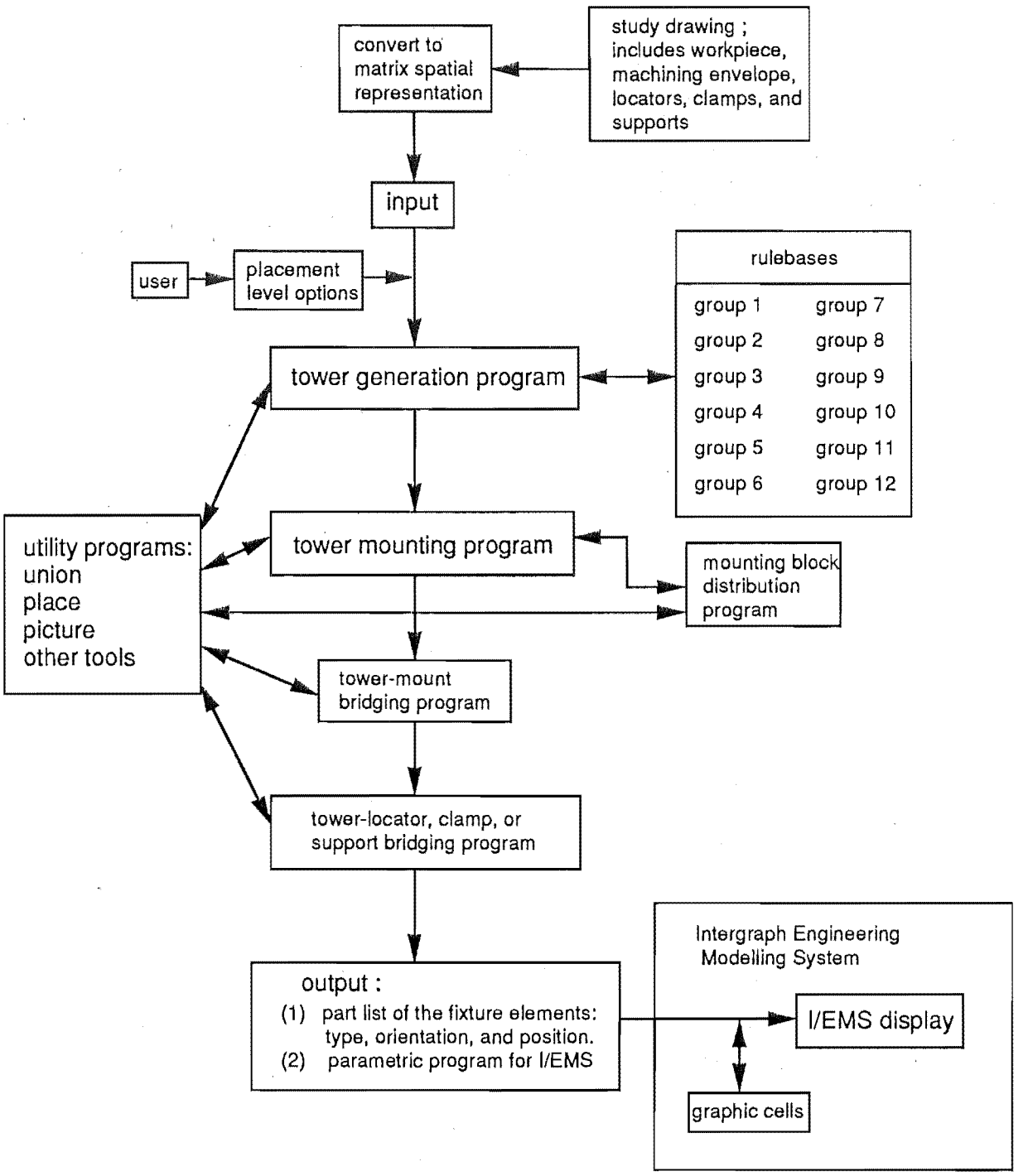


Figure 13 Overall structure of the program

figure 44 and figure 45 respectively. The workpiece is shown in red, and the locators and supports are shown in blue, and the clamps are shown in green.

5.2.2 The placement level

The next step is to choose a suitable placement level for the input above the baseplate. The aim is to place the input so that it allows maximum space for the towers to be built and mounted.

The system allows the user to decide the placement level of the input with respect to the baseplate. To aid the user, three preselected options for the placement levels are available. The selection of the appropriate placement level is dependent on the configuration of the input i.e. the geometry of the workpiece and the machining envelope, and their position relative to the locators, clamps, and supports. However sometimes the appropriate placement may not be obvious, and a different option has to be tested to determine the most suitable one.

The preselected options are :

(1) Placing the lowest point of the locators, clamps, and supports at the design level.

(2) Placing the lowest point of the input at the design level.

(3) Placing the lowest point of the input at the base level. All three options are illustrated in figure 14.

Design level is the level above the baseplate that is equal to the width of the modular blocks, while the base level refers to the surface of the

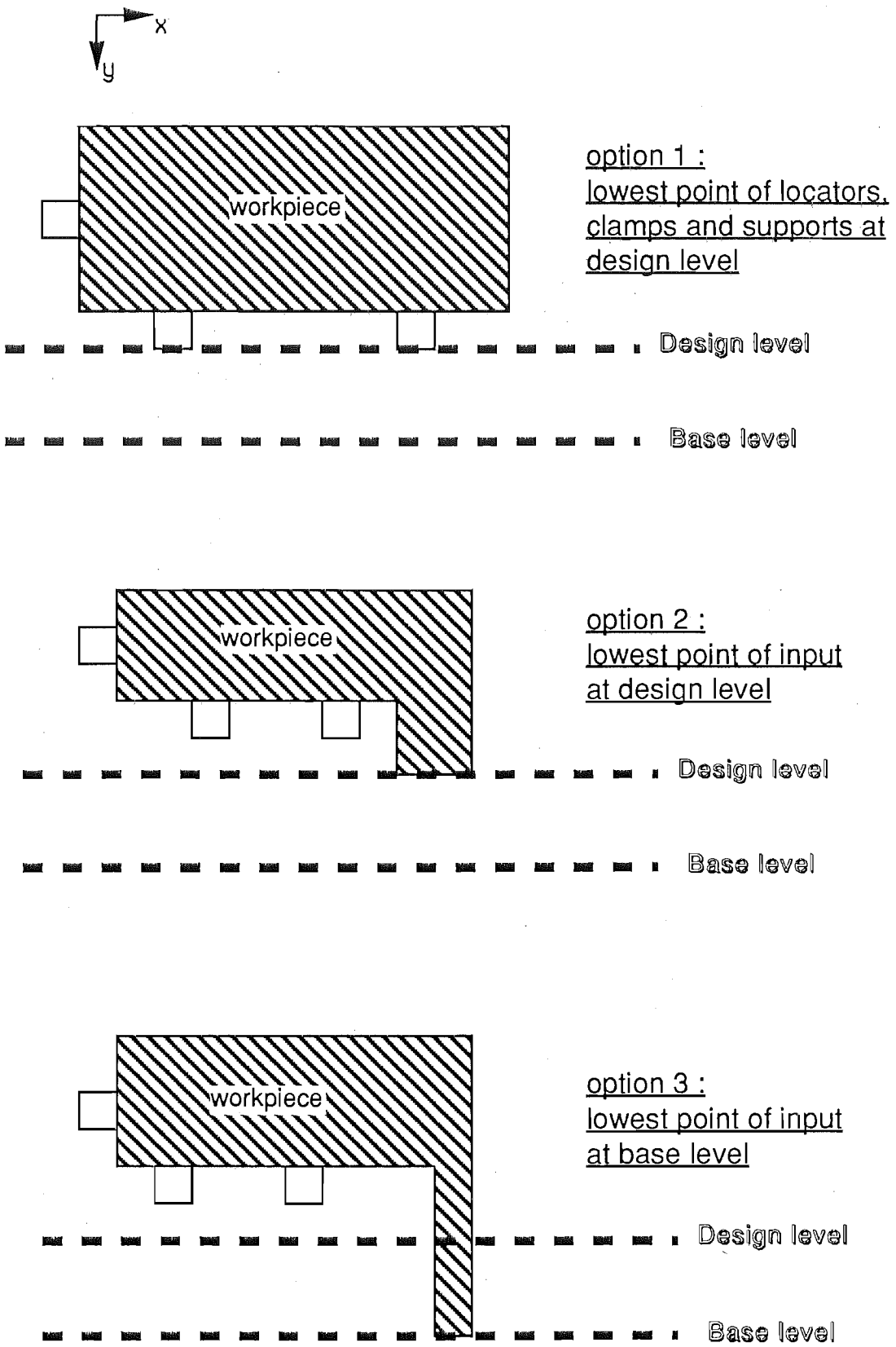


Figure 14 Placement level options

baseplate. Design level is a critical level because any horizontal face locator, clamp or support placed at this level is held by a more flexible tower (tower A), whereas any locator, clamp or support placed above this level has to be held by a more inflexible tower (tower B).

5.2.3 Fixture construction

After selecting a particular height for the workpiece, relative to baseplate, the spatial positioning of the part is unconstrained. Towers are then built by activating the tower generation program. The tower generation program is discussed in chapter 6.

When the process of tower construction is completed, the thrust tower has to be constrained to the baseplate while the remainder ('workpiece unit') is unconstrained. The 'workpiece unit' now consists of the workpiece, the machining envelope, the locators, clamps, and supports and their associated towers of blocks (excluding tower D). This assembly is floating at a fixed height above the baseplate, but not yet located in the horizontal plane.

The tower mounting program can only be activated if the 'workpiece unit' is positioned with respect to the baseplate. Since the number of possible positions to be tested is infinite, the program automatically analyse the positions of all the towers and generates a set of 'advantageous positions'. An advantageous position is a position where at least one tower of the 'workpiece unit' can be directly mounted. The program then places the 'workpiece unit' at each advantageous coordinate and activates the tower mounting program.

At each advantageous position, the towers generated by the tower generation program are positioned on the baseplate unmounted. The tower mounting program has to mount these towers onto the baseplate. Some towers are mounted directly and some indirectly, i.e. through another tower or a mounting block. The tower mounting program is discussed in chapter 7.

The bridging blocks are then constructed to bridge the gap between towers and the mounts, and towers and vertical face locators, clamps, and supports.

The bridging block between the tower and the mount is constructed by activating the *tower-mount bridging program*, and the bridging block between tower and vertical face locators, clamps and support is constructed by activating the *tower-locator bridging program*. These programs are discussed in chapter 7.

Finally the program automatically selects a suitable baseplate. The locators, clamps, and supports are attached to the towers, and the towers are attached to their assigned mounting blocks. The whole assembly is anchored to the baseplate together with those towers which are mounted directly.

5.2.4 Output

The tower mounting program is repeatedly activated until all the advantageous coordinates have been tested. The solution(s), with the maximum number of towers directly mounted is selected. The output of the program gives the type, placement points, and orientation of the

various fixturing elements. With the above information, an assembly drawing of the fixture is generated automatically on a graphics workstation.

CHAPTER 6

TOWER GENERATION

The tower generation program is represented schematically in figure 15. Since the procedures for the generation of different types of tower is basically similar, only one such procedure is shown in the figure.

6.1 TOWER GENERATION PROGRAM

6.1.1 Horizontal face general tower

The tower generation program starts by searching and identifying the horizontal face locators, clamps, and supports above the design level. The program automatically determines the coordinate of the locator, clamp, or support. The process is repeated until all horizontal face locators, clamps, and supports above the design level are detected, and their coordinates recorded.

The program then retrieves the coordinate of each locator, clamp, or support, and derives the height and the position of the horizontal face general tower required. A unique spatial number is assigned to the tower, and matrix representation of the tower is generated. The tower is then constructed to bridge the gap between the reference point on the horizontal face locator, clamp, or support and the baseplate. After constructing the tower, the program automatically searches and identifies any intersection between the tower and the input, and the tower and other towers. If any intersection occurs, the program

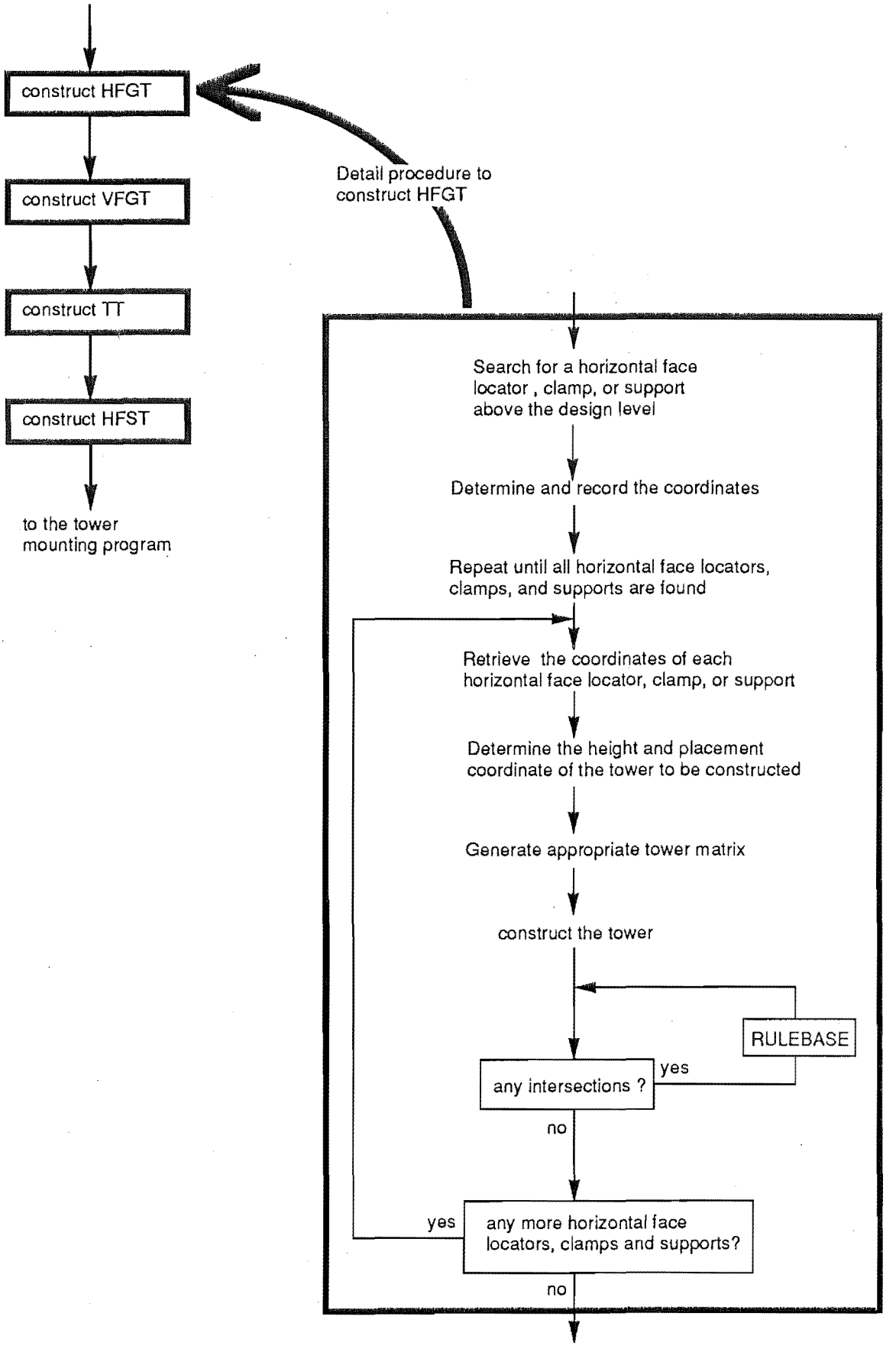


Figure 15 Tower generation program

consults the rule-base for appropriate actions to be taken; which may involve re-positioning, removing, updating or combining the towers. The rulebase is discussed in detail in the next section.

When the tower construction process for each locator, clamp, or support is completed, the program retrieves the next locator, clamp, or support and the tower construction process is repeated until all horizontal face locators, clamps, and supports above the design level have a tower.

6.1.2 Vertical face general tower

The vertical face locator, clamp, or support (including the thrust locator) may be facing north, south, east or west. The towers holding these locators are in turn divided into north, south, east and west as illustrated in figure 16.

The program constructs the towers for the vertical face locators, clamps, and supports in the sequence : north, south, east, and west.

After constructing the horizontal face general tower, the program searches and identifies the vertical face locators, clamps, and supports (north facing). Similarly the program determines and records the coordinate of all vertical face locators, clamps, and supports (north facing).

Using the coordinates of each locator, clamp, or support, the program determines the height and the placement point of the vertical face general tower (north) to be added. A matrix representation of the tower, which contains a unique spatial number assigned to the tower,

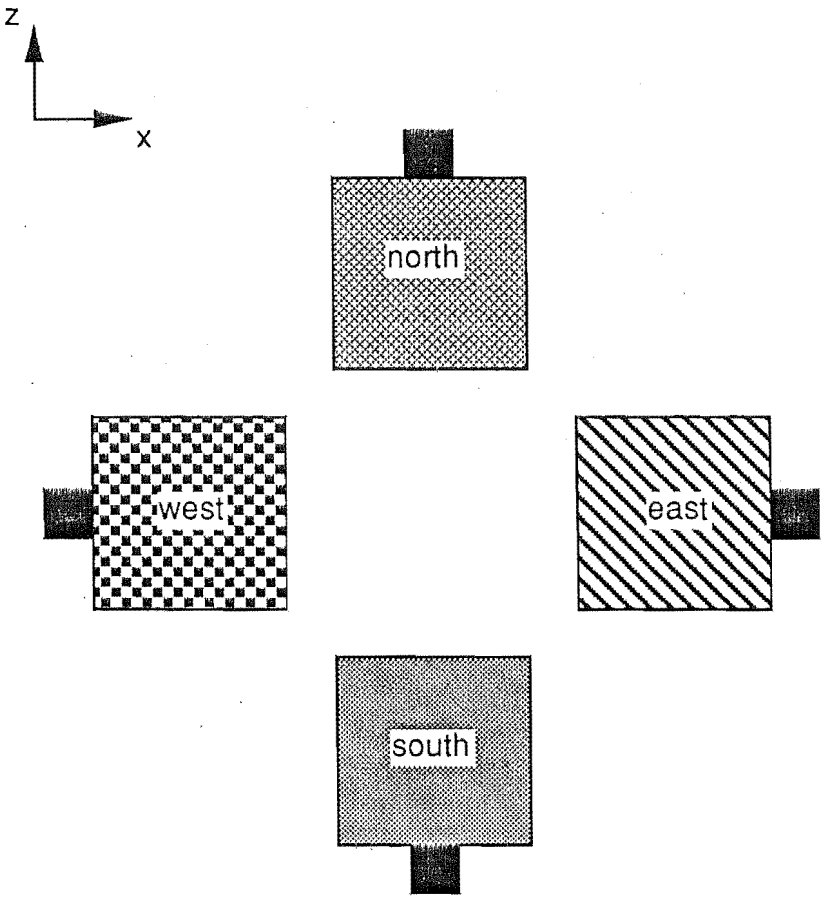


Figure 16 The four possible orientations of tower C/D (plan view)

is generated. The tower is then positioned to bridge the gap between the reference point on the vertical face locator, clamp, or support and the baseplate. If any intersection is detected, the program consults the rule-base for appropriate actions to be taken. When the tower construction process for each locator, clamp, or support is completed, the program retrieves the next locator, clamp, or support and the tower construction process is repeated until all vertical face locators, clamps, and supports (north facing) has a tower.

The tower construction process is repeated for the south facing vertical face locators, clamps, and supports, the east facing vertical face locators, clamps, and supports, and finally the west facing vertical face locators, clamps and supports.

6.1.3 Thrust tower

Next, the thrust tower is constructed to hold the thrust locator (a special case of vertical face locator). The program automatically identifies the thrust locator (north facing), and determines and records the coordinate of the thrust locator. The process is repeated until all thrust locators (north facing) are detected and their coordinates recorded.

The program then retrieves the coordinate of each locator and determines the height and the position of the thrust tower (north) required. A unique spatial number is assigned to the tower, and matrix representation of the tower is generated. The tower is then constructed to bridge the gap between the reference point on the thrust locator and the baseplate, and to give direct connections

between the thrust locators and the baseplate. The direct mounting results in a more rigid structure, and hence able to withstand higher forces. After constructing the tower, the program automatically searches and identifies any intersection between the tower and the input, and the tower and other towers. If any intersection occurs, the program consults the rule-base for appropriate actions to be taken. When the tower construction process for each locator is completed, the program retrieves the next locator, and the tower construction process is repeated until all thrust locators (north facing) has a tower.

The tower construction process is repeated for the south facing thrust locators, the east facing thrust locators, and finally the west facing thrust locators.

6.1.4 Horizontal face special tower

Finally, the program searches and identifies the horizontal face locators, clamps, and supports at the design level, and records their coordinates.

Using the coordinate of each locator, clamp, or support, the program derives the placement coordinate of the horizontal face special tower to be constructed. An object matrix of the tower, which contains a unique spatial number assigned to the tower, is generated. The tower is then positioned to bridge the gap between the reference point on the horizontal face locator, clamp, or support and the baseplate. If any intersection is detected, the program consults the rule-base for appropriate actions to be taken. When the tower construction process for each locator, clamp, or support is completed, the program retrieves

the next locator, clamp, or support and the tower construction process is repeated until all horizontal face locators, clamps, and supports (at the design level) has a tower.

All towers constructed so far are not mounted onto the baseplate. When the process for tower construction is completed, the assembly is ready to be mounted by the tower mounting program which is discussed in the next chapter.

6.2 RULE-BASE

The towers are classified into four basic types, namely, the horizontal face general tower (tower B), the vertical face general tower (tower C), the thrust tower (tower D),and the horizontal face special tower (tower A). In the remainder of the chapter the term 'locator' should be taken to include locator, clamp, and support where applicable.

The rulebase consist of "if.....then" rules governing such intersection. It may move, combine, eliminate and update 'towers'. The rulebase is also able to assimilate new rules to enhance its decision-making capability. The rule-base is always reactivated after any actions on any tower to allow further adjustment if necessary.

The most fundamental rule is that one solid is not allowed to occupy the same space as another solid. Solids includes the workpiece, the machining envelope, the locators, clamps and supports, and some portions of towers.

The rules governing the tower intersections are divided into twelve different groups, and are organised into decision tables (Table 1 to

Table 13). Decision tables are described by Michael Montalbo in the book "Decision Tables" [58] and the description of the decision table are reproduced in the following section.

6.2.1 Decision tables

Decision tables are structured procedure descriptions. They are represented by a rectangle divided into four sections by double lines. The upper left section is called the condition stub. It contains a set of questions covering the entire range of conditions that must be considered before selecting a course of action. The lower left section is called the action stub. It contains the entire set of individual actions governed by the decision table. A course of action is a specification of a set of these individual actions.

The upper right section is called the condition entry. It consists of answers to the questions on the condition stub, the answers along any row corresponding to the questions on that row in the condition stub.

The lower right section is called the action entry. It consists of specifications of which of the actions described in the action stub is to be performed, the specification along any row corresponding to the action described in that row in the action stub.

The association between specific conditions of answers and specific combinations of actions is established by rules, represented by the columns into which the right-hand (or entry) section of the decision table is subdivided. The general interpretation of any rule is as follows:

If the questions asked in the condition stub are answered in the way specified by the condition portion of the rule, then the actions to be performed are those prescribed in the action portion of the rule.

In practice, most tables will contain rules in which some of the questions in the condition stub are ignored. When a square in the condition entry is blank or represented by a dash, it is called a 'Don't-care' square, implying that the course of action prescribed by the rule would be the same whatever answer was placed in the square.

The following interpretations will illustrate the idea:

Last column of table 1

In words, this is the rule : If tower C (west) intersects a solid, and its tower-locator bridging block (TLBB) is clear from interference, then activate 'move tower C (west)' command and reactivate rulebase.

Group 1 tower C vs solid (Table 1)

This group involves the vertical face general tower (tower C) intersecting a solid. The vertical face locator may be facing north, south, east or west. The towers holding these locators are in turn divided into north, south, east and west.

The move C command allows the system to move the tower C away from its locator along the direction of its locator axis, until the tower is clear of the intersection as shown in figure 17. Therefore the tower used to hold these locators would be moved in different directions depending on the direction of the locator. The movement of tower C is continuous.

For the tower to be successful, the bridging block connecting the locator to the tower must not interfere with any solid or any other tower as shown in figure 18.

TABLE 1
GROUP 1 : TOWER C VS SOLID

CONDITION STUB	CONDITION ENTRIES				
Cn vs solid	---	yes	no	no	no
Cs vs solid	---	no	yes	no	no
Ce vs solid	---	no	no	yes	no
Cw vs solid	---	no	no	no	yes
TLBB clear ?	no	yes	yes	yes	yes
ACTION STUB	ACTION ENTRIES				
move Cn		X			
move Cs			X		
move Ce				X	
move Cw					X
fail	X				
Reactivate R/B		X	X	X	X

ABBREVIATIONS	REPRESENTS
Cn	Tower C (north)
Cs	Tower C (south)
Ce	Tower C (east)
Cw	Tower C (west)
TLBB	Tower-locator bridging block
R/B	Rule-base

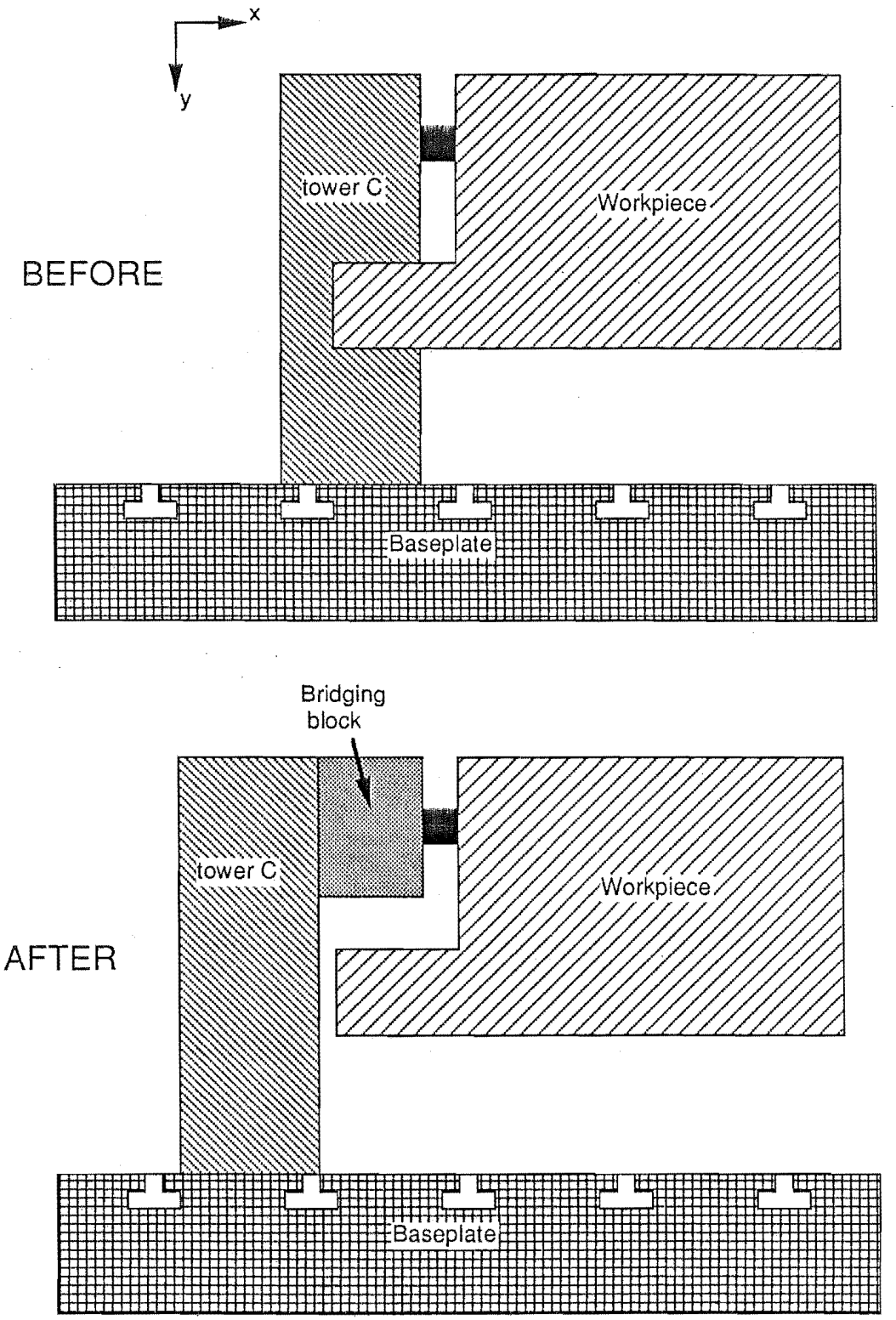


Figure 17 Move C

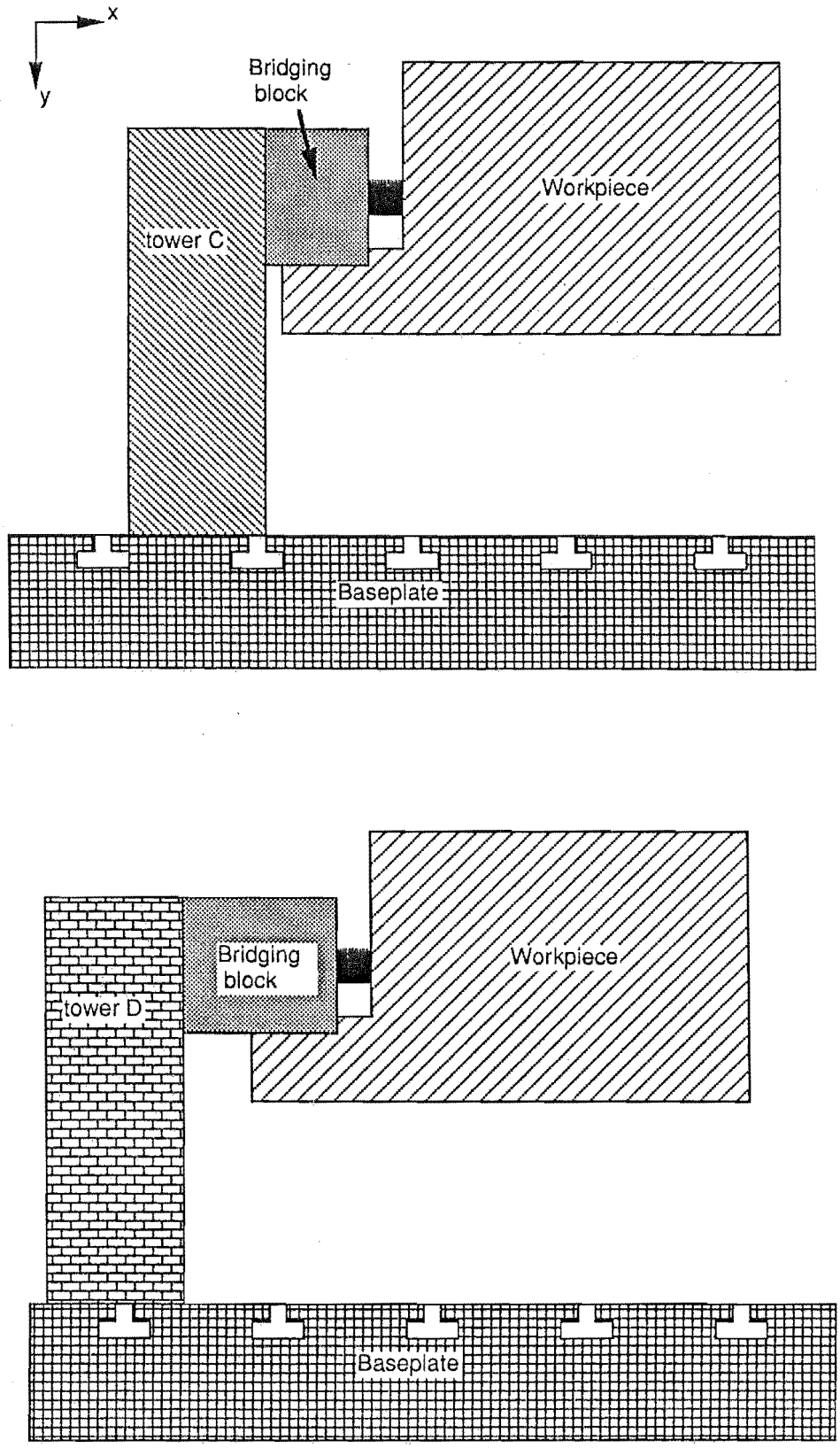


Figure 18 Tower-Location Bridging Block Interference

Group 2 tower D vs solid (Table 2)

This group involves the thrust tower (tower D) intersecting a solid. The thrust locator may be facing north, south, east or west. The towers holding these locators are in turn divided into north, south, east and west.

The move D command allows the system to move the tower D away from its locator along the direction of its locator axis, until the tower is clear of the intersection as shown in figure 19. Therefore the tower used to hold these locators would be moved in different directions depending on the direction of the locator. Unlike tower C, the movement of tower D is discrete not continuous.

For the tower to be successful, the bridging block connecting the locator to the tower must not interfere with any solid or any other tower as shown in figure 18.

TABLE 2
GROUP 2 : TOWER D VS SOLID

CONDITION STUB	CONDITION ENTRIES				
Dn vs solid	---	yes	no	no	no
Ds vs solid	---	no	yes	no	no
De vs solid	---	no	no	yes	no
Dw vs solid	---	no	no	no	yes
TLBB clear ?	no	yes	yes	yes	yes
ACTION STUB	**ACTION ENTRIES**				
move Dn		X			
move Ds			X		
move De				X	
move Dw					X
fail	X				
Reactivate R/B		X	X	X	X

ABBREVIATIONS	REPRESENTS
Dn	Tower D (north)
Ds	Tower D (south)
De	Tower D (east)
Dw	Tower D (west)
TLBB	Tower-locator bridging block
R/B	Rule-base

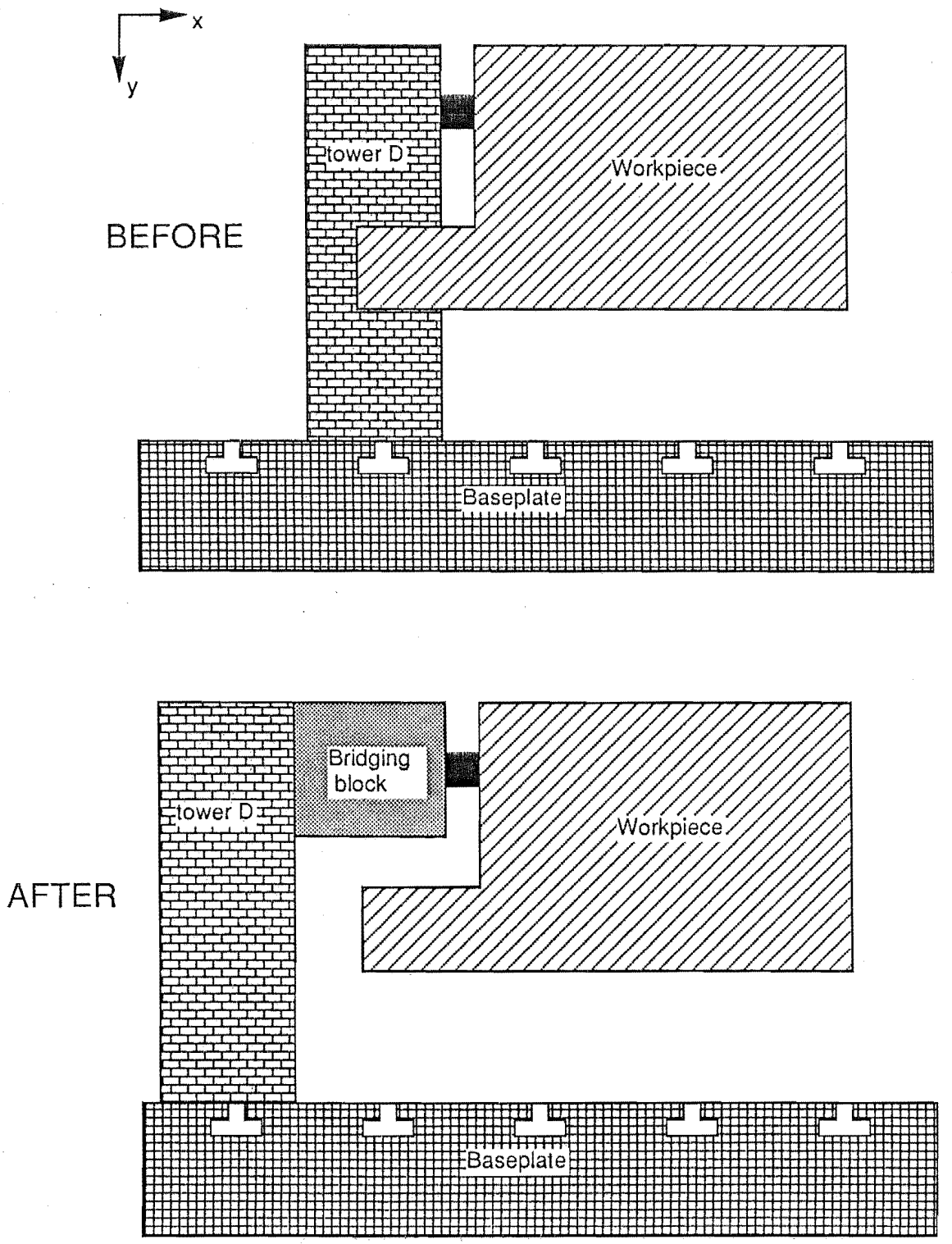


Figure 19 Move D

Group 3 tower C vs tower C - parallel (Table 3)

When two tower Cs are parallel, the directions of their locator axes are the same. When two towers are parallel and aligned, their locator axes are not only parallel but lie on the same plane. The plane is either perpendicular to the x axis or the z axis.

The movement of tower C and the bridging block clearance have been described in group 1. When the above towers intersect, the taller tower is moved to clear any non aligned intersection. However, if the two towers are aligned, they are combined (share tower) to form a combination tower.

The share C command allows the system to replace the intersecting towers by a combination tower holding both locators. The combination tower has the same height as the taller tower, and takes the same position as the tower further away from the locator as illustrated in figure 20.

TABLE 3
GROUP 3 : TOWER C VS TOWER C
(PARALLEL)

CONDITION STUB	CONDITION ENTRIES												
Cn1 vs Cn2	---	yes	yes	yes	no	no	no	no	no	no	no	no	no
Cs1 vs Cs2	---	no	no	no	yes	yes	yes	no	no	no	no	no	no
Ce1 vs Ce2	---	no	no	no	no	no	no	yes	yes	yes	no	no	no
Cw1 vs Cw2	---	no	no	no	no	no	no	no	no	no	yes	yes	yes
Aligned ?	---	yes	no	no	yes	no	no	yes	no	no	yes	no	no
TLBB clear ?	no	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
Cn1 taller ?	---	---	yes	no	---	---	---	---	---	---	---	---	---
Cs1 taller ?	---	---	---	---	---	yes	no	---	---	---	---	---	---
Ce1 taller ?	---	---	---	---	---	---	---	---	yes	no	---	---	---
Cw1 taller ?	---	---	---	---	---	---	---	---	---	---	---	yes	no
ACTION STUB	ACTION ENTRIES												
Share Cn		X											
Share Cs					X								
Share Ce								X					
Share Cw											X		
Move Cn1			X										
Move Cn2				X									
Move Cs1						X							
Move Cs2							X						
Move Ce1									X				
Move Ce2										X			
Move Cw1												X	
Move Cw2													X
Fail	X												
Reactivate R/B		X	X	X	X	X	X	X	X	X	X	X	X

ABBREVIATIONS	REPRESENTS
Cn	tower C (north)
Cs	tower C (south)
Ce	tower C (east)
Cw	tower C (west)
TLBB	tower-locator bridging block
R/B	rule-base

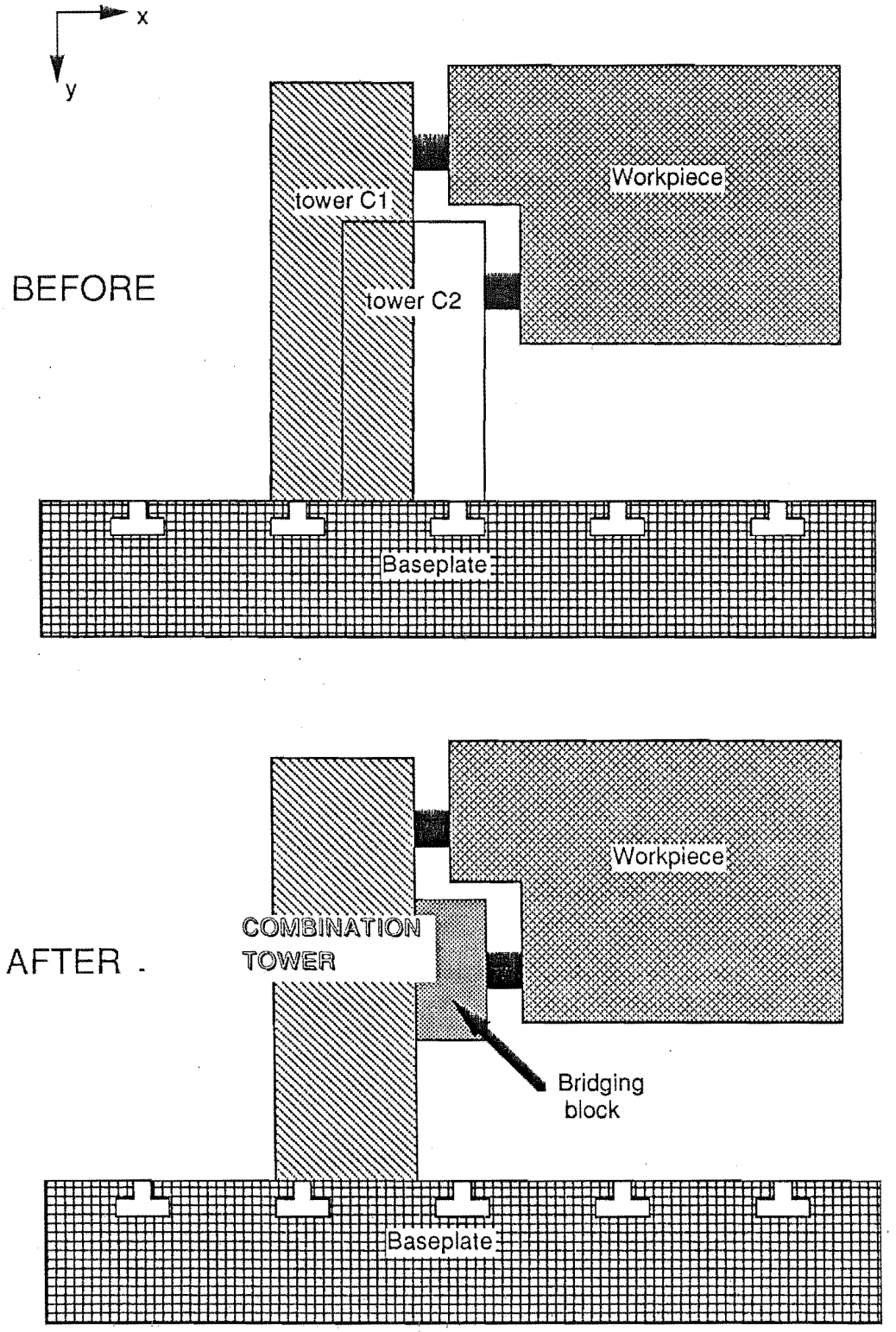


Figure 20 Combination tower of tower C vs tower C (parallel and aligned)

Group 4 tower D vs tower D - parallel (Table 4)

When two tower Ds are parallel, the directions of their locator axis are the same. When two towers are parallel and aligned, their locator axes are not only parallel but lie on the same plane. The plane is either perpendicular to the x axis or the z axis.

The movement of tower D and the bridging block clearance have been described in group 2. When the above towers intersect, the taller tower is moved to clear any non-aligned intersection. However, if the two towers are aligned, they are combined (share tower) to form a combination tower.

The share D command allows the system to replace the intersecting towers by a combination tower holding both locators. The combination tower has the same height as the taller tower, and takes the same position as the tower further away from the locator as illustrated in figure 21.

TABLE 4
GROUP 4 : TOWER D VS TOWER D
(PARALLEL)

CONDITION STUB	CONDITION ENTRIES												
Dn1 vs Dn2	---	yes	yes	yes	no	no	no	no	no	no	no	no	no
Ds1 vs Ds2	---	no	no	no	yes	yes	yes	no	no	no	no	no	no
De1 vs De2	---	no	no	no	no	no	no	yes	yes	yes	no	no	no
Dw1 vs Dw2	---	no	no	no	no	no	no	no	no	no	yes	yes	yes
Aligned ?	---	yes	no	no	yes	no	no	yes	no	no	yes	no	no
TLBB clear ?	no	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
Dn1 taller ?	---	---	yes	no	---	---	---	---	---	---	---	---	---
Ds1 taller ?	---	---	---	---	---	yes	no	---	---	---	---	---	---
De1 taller ?	---	---	---	---	---	---	---	---	yes	no	---	---	---
Dw1 taller ?	---	---	---	---	---	---	---	---	---	---	---	yes	no
ACTION STUB	ACTION ENTRIES												
Share Dn		X											
Share Ds					X								
Share De								X					
Share Dw											X		
Move Dn1			X										
Move Dn2				X									
Move Ds1						X							
Move Ds2							X						
Move De1									X				
Move De2										X			
Move Dw1												X	
Move Dw2													X
Fail	X												
Reactivate R/B		X	X	X	X	X	X	X	X	X	X	X	X

ABBREVIATIONS	REPRESENTS
Dn	tower D (north)
Ds	tower D (south)
De	tower D (east)
Dw	tower D (west)
TLBB	tower-locator bridging block
R/B	rule-base

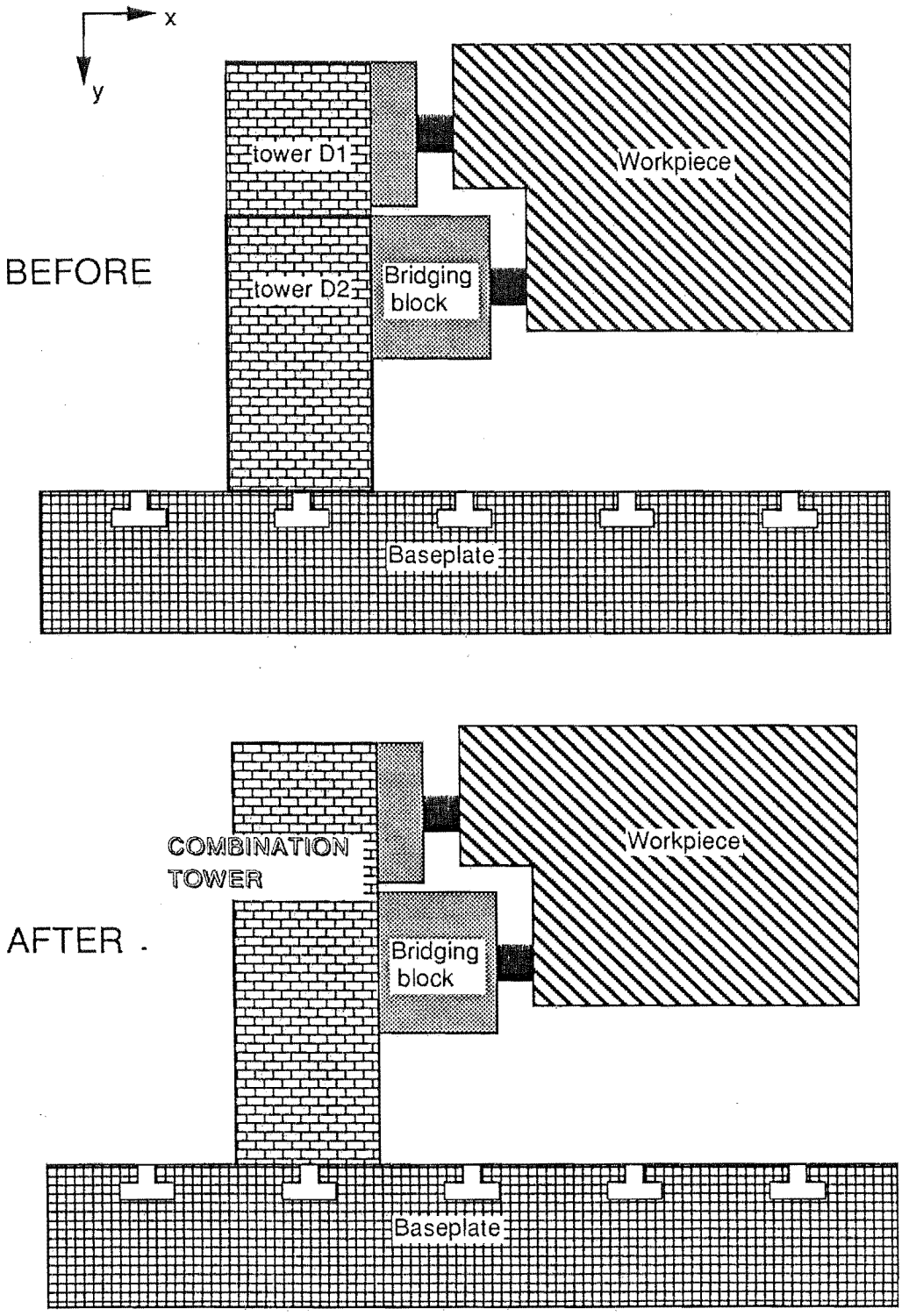


Figure 21 Combination tower of tower D vs tower D (parallel and aligned)

Group 5 tower C vs tower D - parallel (Table 5)

When a tower C and a tower D are parallel, the directions of their locator axes are the same. When two towers are parallel and aligned, their locator axes are not only parallel but lie on the same plane. The plane is either perpendicular to the x axis or the z axis.

The movement of the towers and the bridging block clearance have been described in group 1 and group 2. When the above towers intersect, the taller tower is moved to clear any non aligned intersection. However, if the two towers are aligned, they are combined (share tower) to form a combination tower.

The share D/C command allows the system to replace the intersecting towers by a combination tower holding both locators. The combination tower, as illustrated in figure 22, has the same height as the taller tower and is mounted to a slot. The combination tower must not be in front of its intersecting towers.

TABLE 5
GROUP 5 : TOWER D VS TOWER C
(PARALLEL)

CONDITION STUB	CONDITION ENTRIES												
Dn vs Cn	---	yes	yes	yes	no	no	no	no	no	no	no	no	no
Ds vs Cs	---	no	no	no	yes	yes	yes	no	no	no	no	no	no
De vs Ce	---	no	no	no	no	no	no	yes	yes	yes	no	no	no
Dw vs Cw	---	no	no	no	no	no	no	no	no	no	yes	yes	yes
Aligned ?	---	yes	no	no	yes	no	no	yes	no	no	yes	no	no
TLBB clear ?	no	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
Dn taller ?	---	---	yes	no	---	---	---	---	---	---	---	---	---
Ds taller ?	---	---	---	---	---	yes	no	---	---	---	---	---	---
De taller ?	---	---	---	---	---	---	---	---	yes	no	---	---	---
Dw taller ?	---	---	---	---	---	---	---	---	---	---	---	yes	no
ACTION STUB	ACTION ENTRIES												
Share Dn/Cn		X											
Share Ds/Cs					X								
Share De/Ce								X					
Share Dw/Cw											X		
Move Dn			X										
Move Cn				X									
Move Ds						X							
Move Cs							X						
Move De									X				
Move Ce										X			
Move Dw												X	
Move Cw													X
Fail	X												
Reactivate R/B		X	X	X	X	X	X	X	X	X	X	X	X

ABBREVIATIONS	REPRESENTS
Cn	tower C (north)
Cs	tower C (south)
Ce	tower C (east)
Cw	tower C (west)
Dn	tower D (north)
Ds	tower D (south)
De	tower D (east)
Dw	tower D (west)
TLBB	tower-locator bridging block
R/B	rule-base

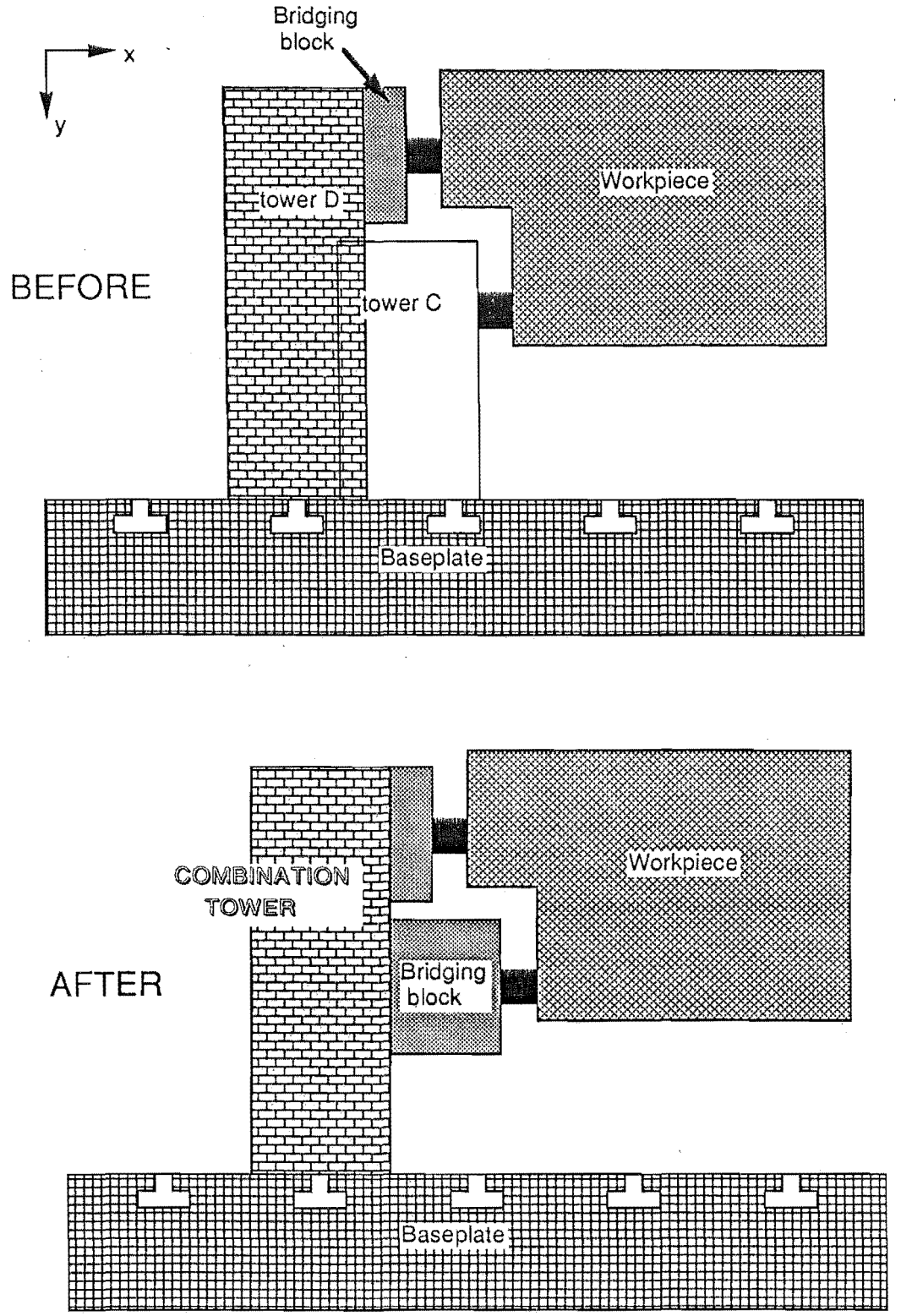


Figure 22 Combination tower of tower C vs tower D (parallel and aligned)

Group 6 tower C vs tower C - perpendicular (Table 6)

When two tower Cs are perpendicular their locator axes are at right angles to each other. The movement of tower C and the bridging block clearance have been described in group 1. When the above towers intersect, the taller tower is moved to clear any intersection. However, sometimes the two towers may be combined (share tower) to form a combination tower.

The share C command allows the system to replace the intersecting towers by a combination tower holding both locators. The combination tower, as illustrated in figure 23, has the same height as the taller tower. The center of the combination tower lies at the intersection of the two locator axes.

The relative position of the towers will determine whether the share tower or the move tower command should be activated. If you assume the direction of the axes is positive, then the center of both blocks can be defined. If these are not negative (Case A), then the share tower command is activated, as shown in figure 23. However, if the center of any one tower is negative (Case B), then the move tower command is activated, as shown in figure 24.

TABLE 6
GROUP 6 : TOWER C VS TOWER C
(PERPENDICULAR)

CONDITION STUB	CONDITION ENTRIES												
Cn vs Ce	---	yes	no	no	no	yes	no	no	no	yes	no	no	no
Cn vs Cw	---	no	yes	no	no	no	yes	no	no	no	no	yes	no
Cs vs Ce	---	no	no	yes	no	no	no	yes	no	no	yes	no	no
Cs vs Cw	---	no	no	no	yes	no	no	no	yes	no	no	no	yes
Case A	---	yes	yes	yes	yes	no	no	no	no	no	no	no	no
Case B	---	no	no	no	no	yes	yes	yes	yes	yes	yes	yes	yes
TLBB clear ?	no	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
Cn taller ?	---	---	---	---	---	yes	yes	---	---	---	---	---	---
Cs taller ?	---	---	---	---	---	---	---	yes	yes	---	---	---	---
Ce taller ?	---	---	---	---	---	---	---	---	---	yes	yes	---	---
Cw taller ?	---	---	---	---	---	---	---	---	---	---	---	yes	yes
ACTION STUB	ACTION ENTRIES												
Share Cn/Ce		X											
Share Cn/Cw			X										
Share Cs/Ce				X									
Share Cs/Cw					X								
Move Cn						X	X						
Move Cs								X	X				
Move Ce										X	X		
Move Cw												X	X
Fail	X												
Reactivate R/B		X	X	X	X	X	X	X	X	X	X	X	X

ABBREVIATIONS	REPRESENTS
Cn	tower C (north)
Cs	tower C (south)
Ce	tower C (east)
Cw	tower C (west)
TLBB	tower-locator bridging block
R/B	rule-base

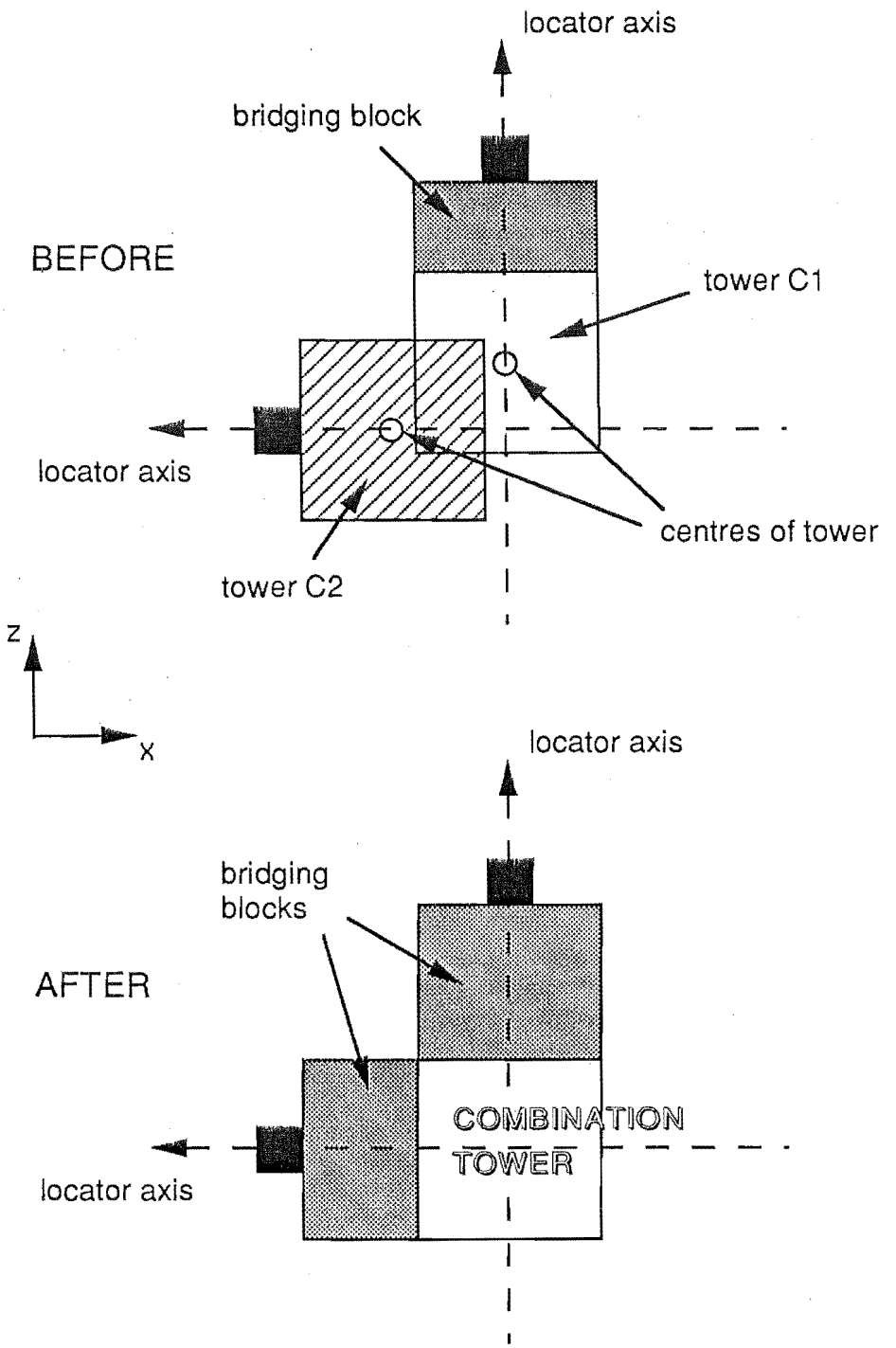


Figure 23 Tower C vs Tower C (perpendicular) - Case A

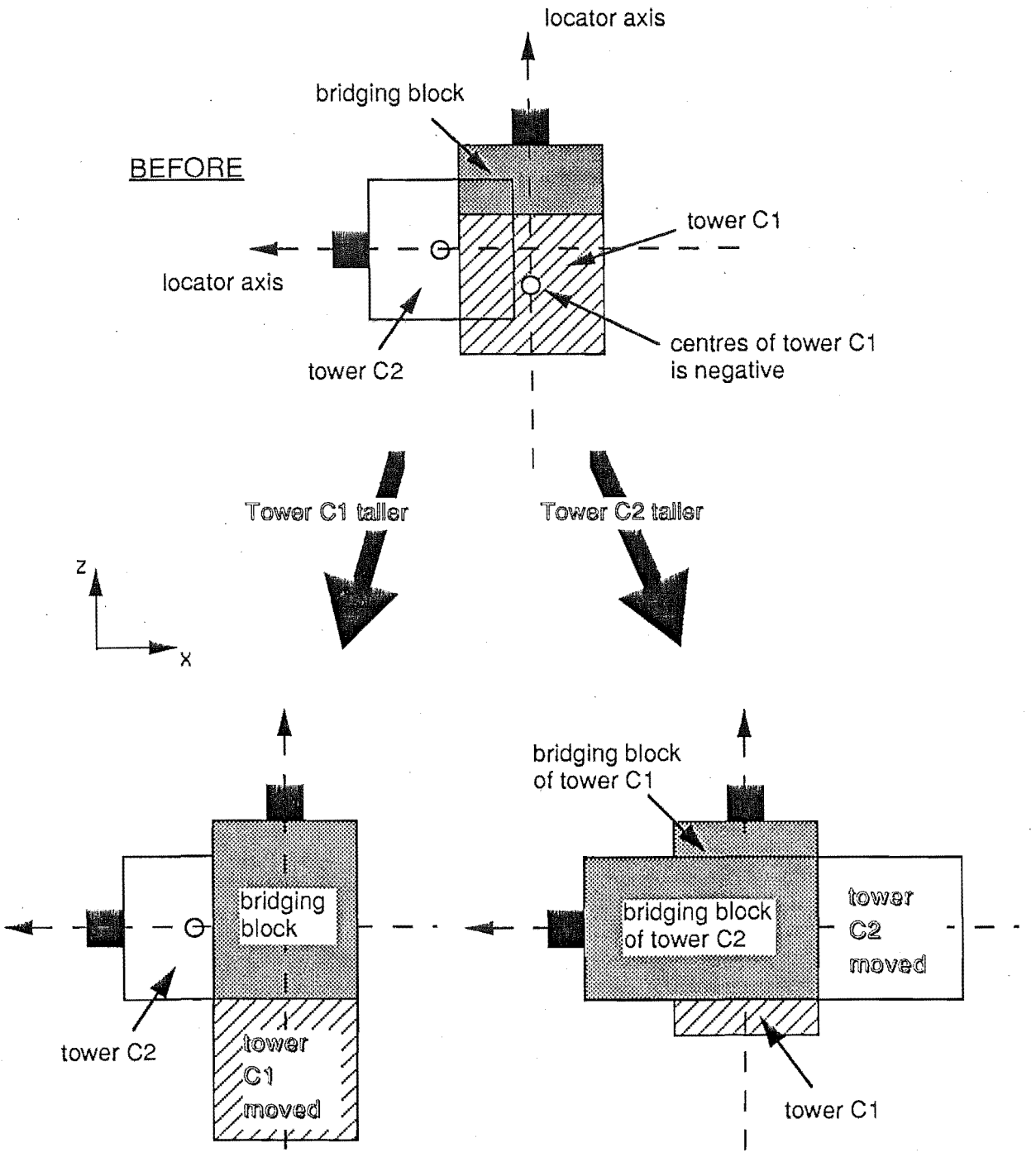


Figure 24 Tower C vs Tower C (perpendicular) - Case B

Group 7 tower D vs tower D - perpendicular (Table 7)

When two tower Ds are perpendicular, their locator axes are at right angles to each other. The movement of tower D and the bridging block clearance have been described in group 2. When the above towers intersect, the taller tower is moved to clear any intersection.

Group 8 tower C vs tower D - perpendicular (Table 8)

When a tower C and a tower D are perpendicular, their locator axes are at right angles to each other. The movement of the towers and the bridging block clearance have been described in group 1 and group 2. When the above towers intersect, the taller tower is moved to clear any intersection.

TABLE 7
GROUP 7 : TOWER D VS TOWER D
(PERPENDICULAR)

CONDITION STUB	CONDITION ENTRIES								
Dn vs De	---	yes	no	no	no	yes	no	no	no
Dn vs Dw	---	no	yes	no	no	no	yes	no	no
Ds vs De	---	no	no	yes	no	no	no	yes	no
Ds vs Dw	---	no	no	no	yes	no	no	no	yes
TLBB clear ?	no	yes	yes	yes	yes	yes	yes	yes	yes
Dn taller ?	---	yes	yes	---	---	no	no	---	---
Ds taller ?	---	---	---	yes	yes	---	---	no	no
ACTION STUB	ACTION ENTRIES								
Move Dn		X	X						
Move Ds				X	X				
Move De						X		X	
Move Dw							X		X
Fail	X								
Reactivate R/B		X	X	X	X	X	X	X	X

ABBREVIATIONS	REPRESENTS
Dn	tower D (north)
Ds	tower D (south)
De	tower D (east)
Dw	tower D (west)
TLBB	tower-locator bridging block
R/B	rule-base

TABLE 8
GROUP 8 : TOWER D VS TOWER C
(PERPENDICULAR)

CONDITION STUB	CONDITION ENTRIES																
Dn vs Ce	--	Y	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N
Dn vs Cw	--	N	Y	N	N	N	N	N	N	N	Y	N	N	N	N	N	N
Ds vs Ce	--	N	N	Y	N	N	N	N	N	N	N	Y	N	N	N	N	N
Ds vs Cw	--	N	N	N	Y	N	N	N	N	N	N	N	Y	N	N	N	N
De vs Cn	--	N	N	N	N	Y	N	N	N	N	N	N	N	Y	N	N	N
De vs Cs	--	N	N	N	N	N	Y	N	N	N	N	N	N	N	Y	N	N
Dw vs Cn	--	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	Y	N
Dw vs Cs	--	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	Y
TLBB clear ?	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Dn taller ?	--	Y	Y	--	--	--	--	--	--	--	--	--	--	--	--	--	--
Ds taller ?	--	--	--	Y	Y	--	--	--	--	--	--	--	--	--	--	--	--
De taller ?	--	--	--	--	--	Y	Y	--	--	--	--	--	--	--	--	--	--
Dw taller ?	--	--	--	--	--	--	--	Y	Y	--	--	--	--	--	--	--	--
Cn taller ?	--	--	--	--	--	--	--	--	--	--	--	--	--	Y	--	Y	--
Cs taller ?	--	--	--	--	--	--	--	--	--	--	--	--	--	--	Y	--	Y
Ce taller ?	--	--	--	--	--	--	--	--	--	Y	--	Y	--	--	--	--	--
Cw taller ?	--	--	--	--	--	--	--	--	--	--	Y	--	Y	--	--	--	--
ACTION STUB	ACTION ENTRIES																
Move Dn		X	X														
Move Ds				X	X												
Move De						X	X										
Move Dw								X	X								
Move Cn														X		X	
Move Cs															X		X
Move Ce										X		X					
Move Cw											X		X				
Fail	X																
React. R/B		X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ABBREVIATIONS	REPRESENTS
Cn	tower C (north)
Cs	tower C (south)
Ce	tower C (east)
Cw	tower C (west)
Dn	tower D (north)
Ds	tower D (south)
De	tower D (east)
Dw	tower D (west)
Y	yes
N	no
TLBB	tower-locator bridging block
React. R/B	Reactivate rule-base

Group 9 tower C vs tower B (Table 9)

The movement of tower C and the bridging block clearance have been described in group 1. Since tower B is holding a horizontal face locator, which does not allow its tower to be moved, tower C is moved to clear any non aligned intersection. However, if the two towers are aligned, they may be combined (share tower) to form a combination tower. (The two towers is aligned if their locator axis intersects).

The share B/C command allows the system to replace the intersecting towers by a combination tower holding both locators. The combination tower, as illustrated in figure 25, has the same height as tower B, and takes the same position as tower B.

Group 10 tower D vs tower B (Table 10)

The movement of tower D and the bridging block clearance have been described in group 2. Since tower B is holding a horizontal face locator, which does not allow its tower to be moved, tower D is moved to clear any non aligned intersection.

TABLE 9
GROUP 9 : TOWER C VS TOWER B

CONDITION STUB	CONDITION ENTRIES												
Cn vs B	---	yes	yes	yes	no	no	no	no	no	no	no	no	no
Cs vs B	---	no	no	no	yes	yes	yes	no	no	no	no	no	no
Ce vs B	---	no	no	no	no	no	no	yes	yes	yes	no	no	no
Cw vs B	---	no	no	no	no	no	no	no	no	no	yes	yes	yes
Aligned ?	---	yes	yes	no	yes	yes	no	yes	yes	no	yes	yes	no
TLBB clear ?	no	yes	---	yes	yes	---	yes	yes	---	yes	yes	---	yes
Cn taller ?	---	no	yes	---	---	---	---	---	---	---	---	---	---
Cs taller ?	---	---	---	---	no	yes	---	---	---	---	---	---	---
Ce taller ?	---	---	---	---	---	---	---	no	yes	---	---	---	---
Cw taller ?	---	---	---	---	---	---	---	---	---	---	no	yes	---
ACTION STUB	ACTION ENTRIES												
Share Cn/B		X											
Share Cs/B					X								
Share Ce/B								X					
Share Cw/B											X		
Move Cn				X									
Move Cs							X						
Move Ce										X			
Move Cw													X
Fail	X		X			X			X			X	
Reactivate R/B		X		X	X		X	X		X	X		X

ABBREVIATIONS	REPRESENTS
B	tower B
Cn	tower C (north)
Cs	tower C (south)
Ce	tower C (east)
Cw	tower C (west)
TLBB	tower-locator bridging block
R/B	rule-base

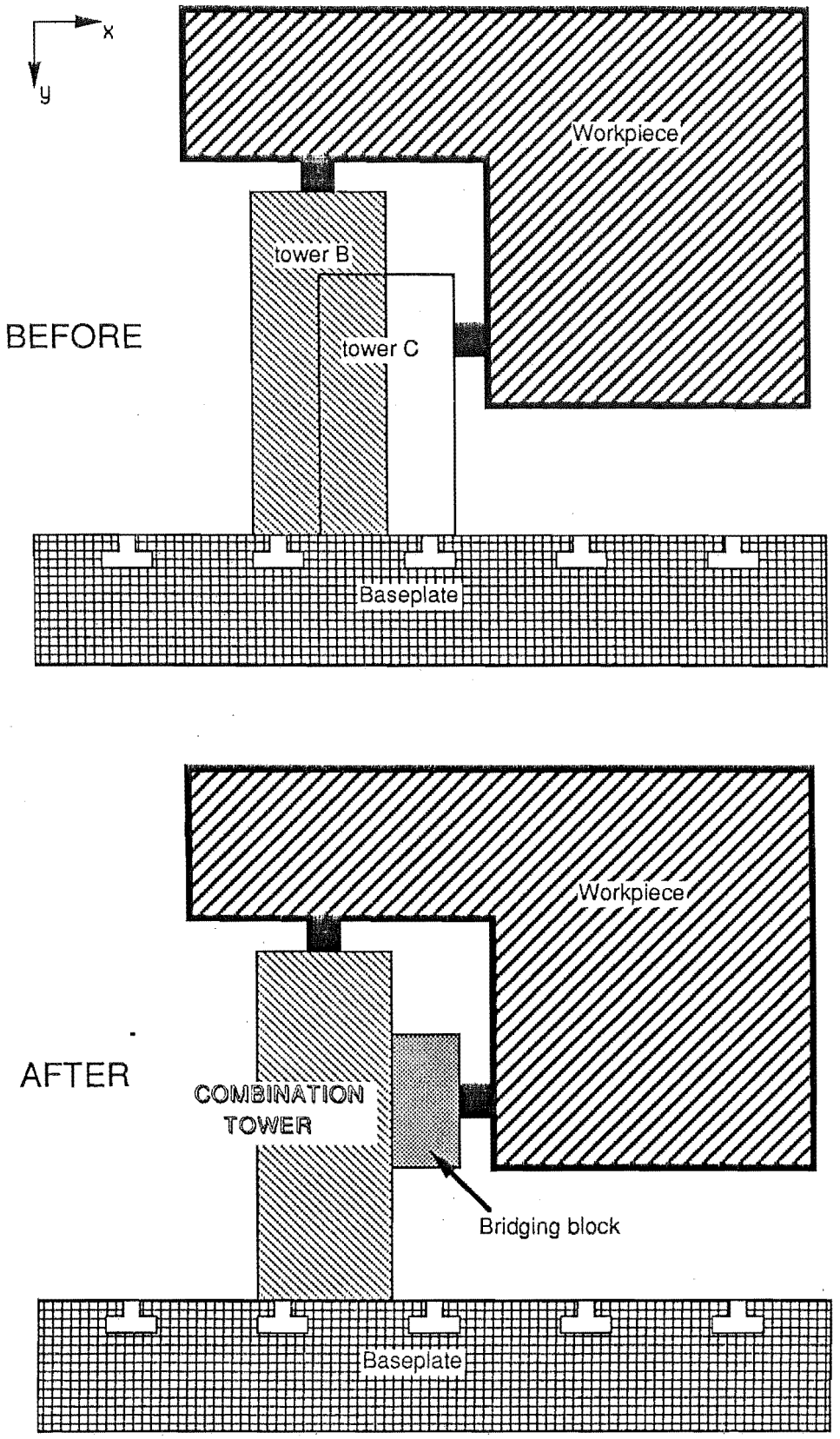


Figure 25 Combination tower of tower B vs tower C (aligned)

TABLE 10
GROUP 10 : TOWER D VS TOWER B

CONDITION STUB	CONDITION ENTRIES				
Dn vs B	---	yes	no	no	no
Ds vs B	---	no	yes	no	no
De vs B	---	no	no	yes	no
Dw vs B	---	no	no	no	yes
TLBB clear ?	no	yes	yes	yes	yes
ACTION STUB	**ACTION ENTRIES**				
move Dn		X			
move Ds			X		
move De				X	
move Dw					X
fail	X				
Reactivate R/B		X	X	X	X

ABBREVIATIONS	REPRESENTS
B	Tower B
Dn	Tower D (north)
Ds	Tower D (south)
De	Tower D (east)
Dw	Tower D (west)
TLBB	Tower-locator bridging block
R/B	Rule-base

Group 11 Tower A vs others (Table 11)

There are two possible orientations for tower A as shown in figure 26. Only one orientation would be chosen during the design process.

The selection of the appropriate orientation is dependent on the space available, relationship with other towers, and the mounting possibilities.

Group 12 tower A vs tower A (Table 12 and Table 13)

When two tower As intersect, there are two possible situations : single sector and double sector intersect, as shown below in figure 26.

In both situations, a decision has to be made whether to update one or the other tower. So instead of updating the tower, each tower A vs tower A intersection is recorded by switching the appropriate tower indicator off. Each tower A has four indicators monitoring the status of its four sectors, namely A1,A2,A3,and A4 as illustrated in figure 27.

If two opposite indicator of the same tower is switched off, the orientation of that tower is finalised as shown in table 13 and illustrated in figure 28.

TABLE 11
GROUP 11 : TOWER A VS OTHERS

CONDITION STUB	CONDITION ENTRIES							
A1 vs others	yes	no	---	no	yes	yes	---	---
A2 vs others	no	yes	no	---	yes	---	yes	---
A3 vs others	---	no	yes	no	---	---	yes	yes
A4 vs others	no	---	no	yes	---	yes	---	yes
ACTION STUB	ACTION ENTRIES							
Orientation ZZ		X		X				
Orientation XX	X		X					
Fail					X	X	X	X
Reactivate R/B	X	X	X	X				

ABBREVIATIONS	REPRESENTS
A1	sector 1 of tower A
A2	sector 2 of tower A
A3	sector 3 of tower A
A4	sector 4 of tower A
others	any other towers or objects
R/B	Rule-base

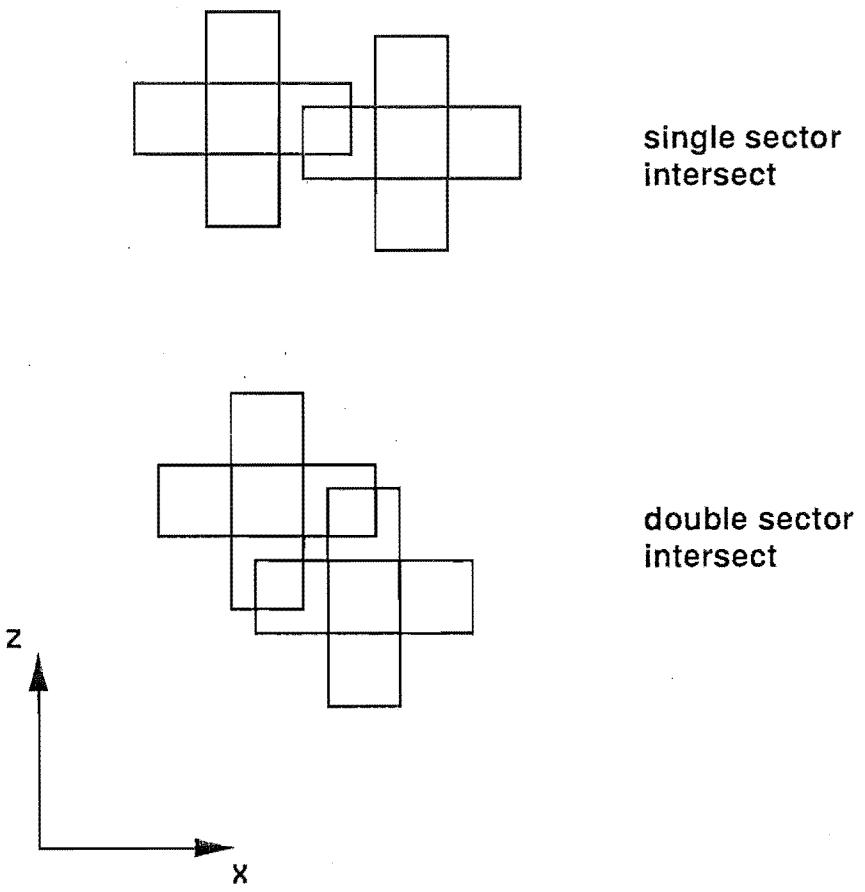


Figure 26 Types of tower A vs tower A intersection

TABLE 12
GROUP 12 : TOWER A VS TOWER A

CONDITION STUB	CONDITION ENTRIES							
A1[1] vs A3[2]	yes	no	no	no	no	no	no	no
A2[1] vs A4[2]	no	yes	no	no	no	no	no	no
A1[2] vs A3[1]	no	no	yes	no	no	no	no	no
A2[2] vs A4[1]	no	no	no	yes	no	no	no	no
A1[1] vs A2[2]	no	no	no	no	yes	no	no	no
A2[1] vs A3[2]	no	no	no	no	no	yes	no	no
A3[1] vs A4[2]	no	no	no	no	no	no	yes	no
A4[1] vs A1[2]	no	no	no	no	no	no	no	yes
A1[2] vs A2[1]	no	no	no	no	no	no	yes	no
A2[2] vs A3[1]	no	no	no	no	no	no	no	yes
A3[2] vs A4[1]	no	no	no	no	yes	no	no	no
A4[2] vs A1[1]	no	no	no	no	no	yes	no	no
ACTION STUB	ACTION ENTRIES							
indicator A1[1]	off				off	off		
indicator A2[1]		off				off	off	
indicator A3[1]			off				off	off
indicator A4[1]				off	off			off
indicator A1[2]			off				off	off
indicator A2[2]				off	off			off
indicator A3[2]	off				off	off		
indicator A4[2]		off				off	off	
Reactivate R/B	X	X	X	X	X	X	X	X

ABBREVIATIONS	REPRESENTS
A1	sector 1 of tower A
A2	sector 2 of tower A
A3	sector 3 of tower A
A4	sector 4 of tower A
R/B	Rule-base

TABLE 13
UPDATING TOWER A

CONDITION STUB	CONDITION ENTIRES					
indicator A1[1]	off	---	off	---	---	---
indicator A2[1]	off	---	---	off	---	---
indicator A3[1]	off	---	off	---	---	---
indicator A4[1]	off	---	---	off	---	---
indicator A1[2]	---	off	---	---	off	---
indicator A2[2]	---	off	---	---	---	off
indicator A3[2]	---	off	---	---	off	---
indicator A4[2]	---	off	---	---	---	off
ACTION STUB	ACTION ENTRIES					
tower A[1] orientation XX				X		
tower A[1] orientation ZZ			X			
tower A[2] orientation XX						X
tower A[2] orientation ZZ					X	
fail	X	X				

ABBREVIATIONS	REPRESENTS
A1	sector 1 of tower A
A2	sector 2 of tower A
A3	sector 3 of tower A
A4	sector 4 of tower A

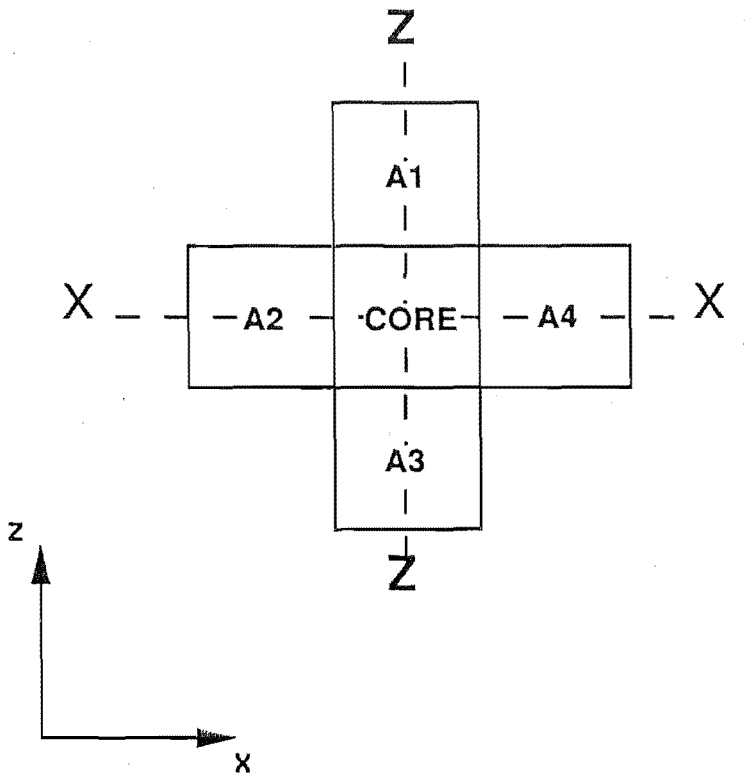


Figure 27 Tower A

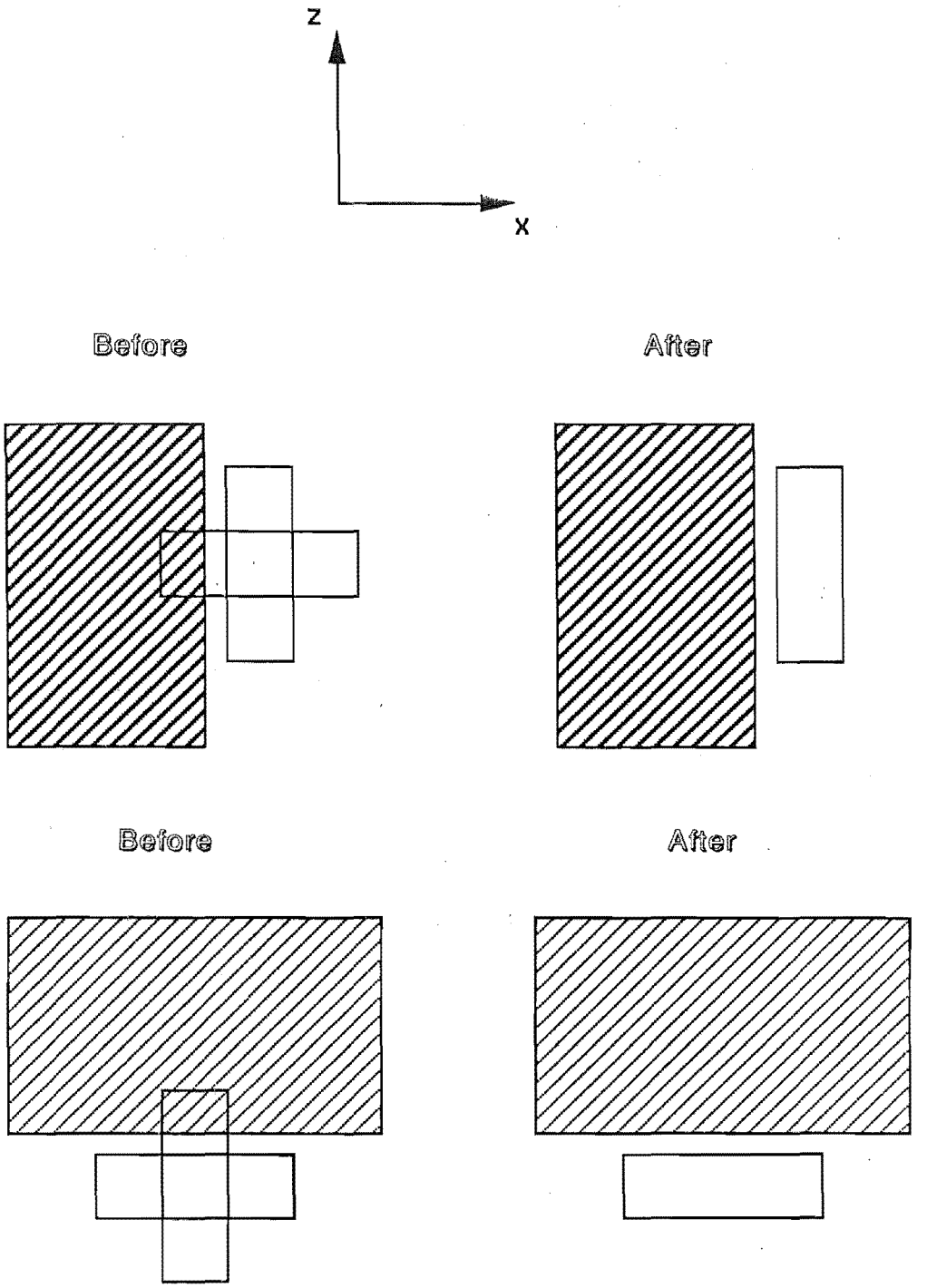


Figure 28 Solid vs tower A

CHAPTER 7

TOWER MOUNTING

7.1 TOWER MOUNTING PROGRAM

After the tower generation program, the inputs and the towers are standing on the baseplate unmounted. Besides the four basic types of towers, namely, the horizontal face general tower, the vertical face general tower, the thrust tower and the horizontal face special tower, there are some combination towers.

When the tower mounting program is activated, it analyses the position of all towers with respect to the baseplate. The tower mounting program, first, mounts the thrust towers directly onto the baseplate. Then, it mounts any other towers that are standing on a slot of the baseplate directly onto the baseplate.

The remaining towers have to be mounted indirectly, i.e. through another tower or a mounting block. If an unmounted tower is aligned with an adjacent mounted tower, the unmounted tower must be fastened to the mounted tower.

If an indirect tower cannot be mounted through another tower, then a mounting block is required. In such cases the tower will have to be mounted through a mounting block distribution program. This program studies the options available to each tower, and assigns the mounting blocks accordingly. The mounting block distribution

program generates, selects, combines and assigns mounting blocks (figure 29).

7.1.1 Mounting block distribution program

Generating the mounting blocks

Since the mounting blocks are mounted onto the slots of the baseplate and occupy the space between the design level and the baseplate, they can only intersect the lower block of a tower as shown in figure 30. (The lower block of the tower is connected to the upper block by a 't-bolt'). The mounting block must be placed on a slot of the baseplate nearest to the tower. It is allowed to intersect its own tower, because the lower block of the tower can slide along the head of the 't-bolt' as shown in figure 30, and its length can be adjusted. However, the mounting block is not allowed to interfere with the 't-bolt'. The region of the lower block restricted by the 't-bolt' is called the 'core'of the tower.

Each tower, theoretically, has four possible mounting blocks as shown in figure 31 : north of the tower, south of the tower, east of the tower, west of the tower. Therefore if there are 6 towers to be mounted, the program will generate a total of 24 mounting blocks. However due to the space constraints not all mounting blocks are clear of intersection.

The mounting block distribution program starts by identifying each of those towers that require a mounting block. The program determines and records the position of these towers.

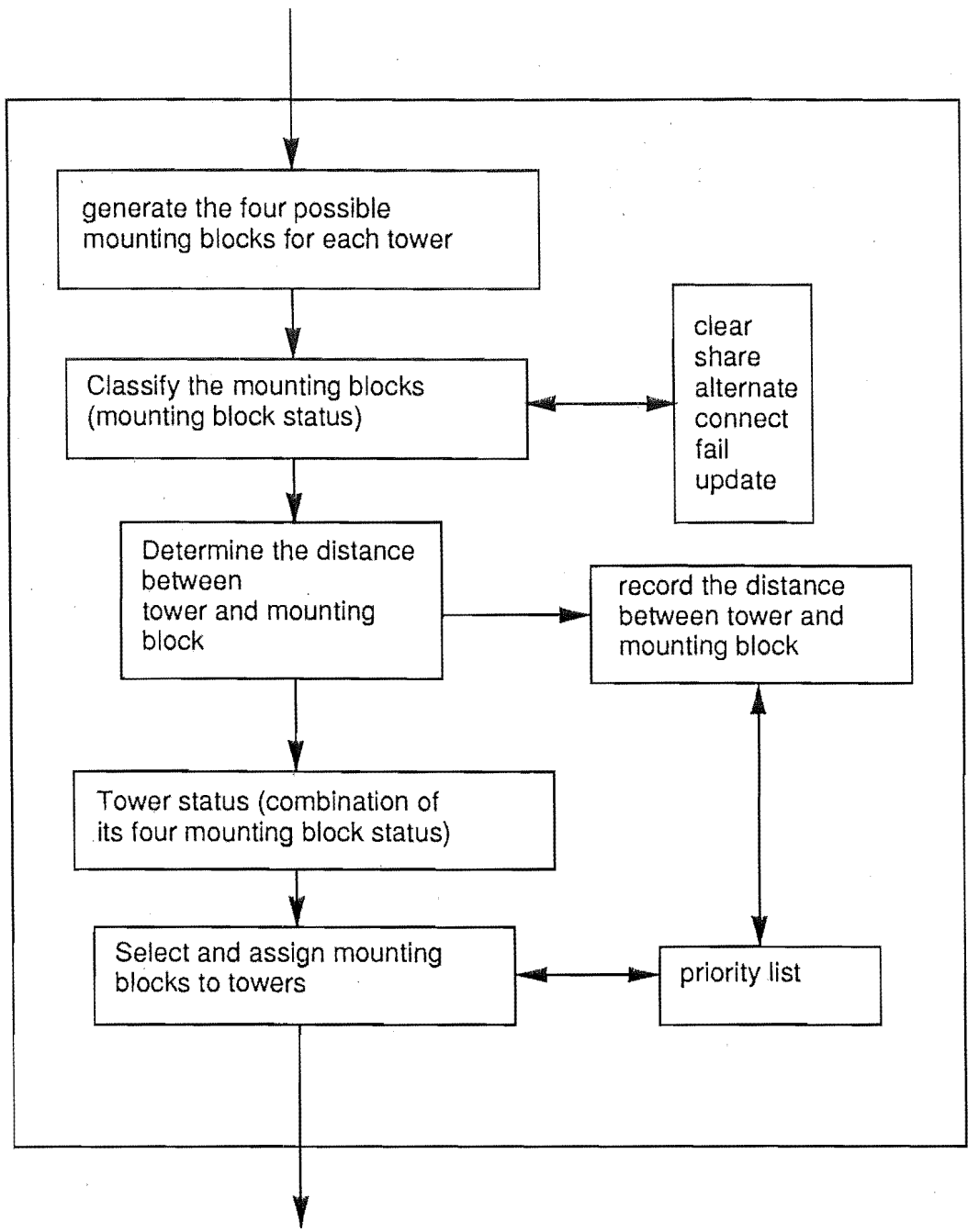


Figure 29 Mounting block distribution program

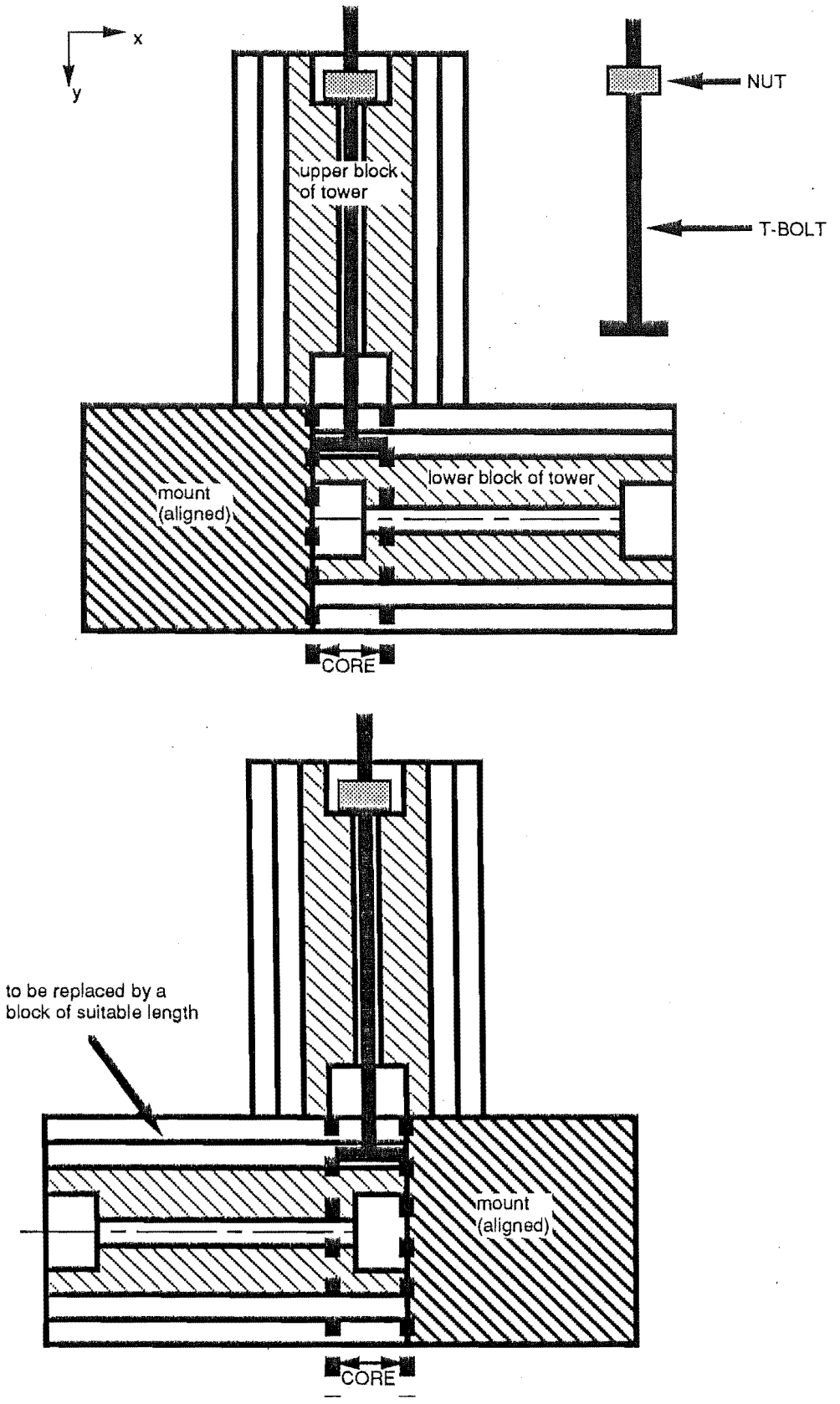


Figure 30 Core of a tower

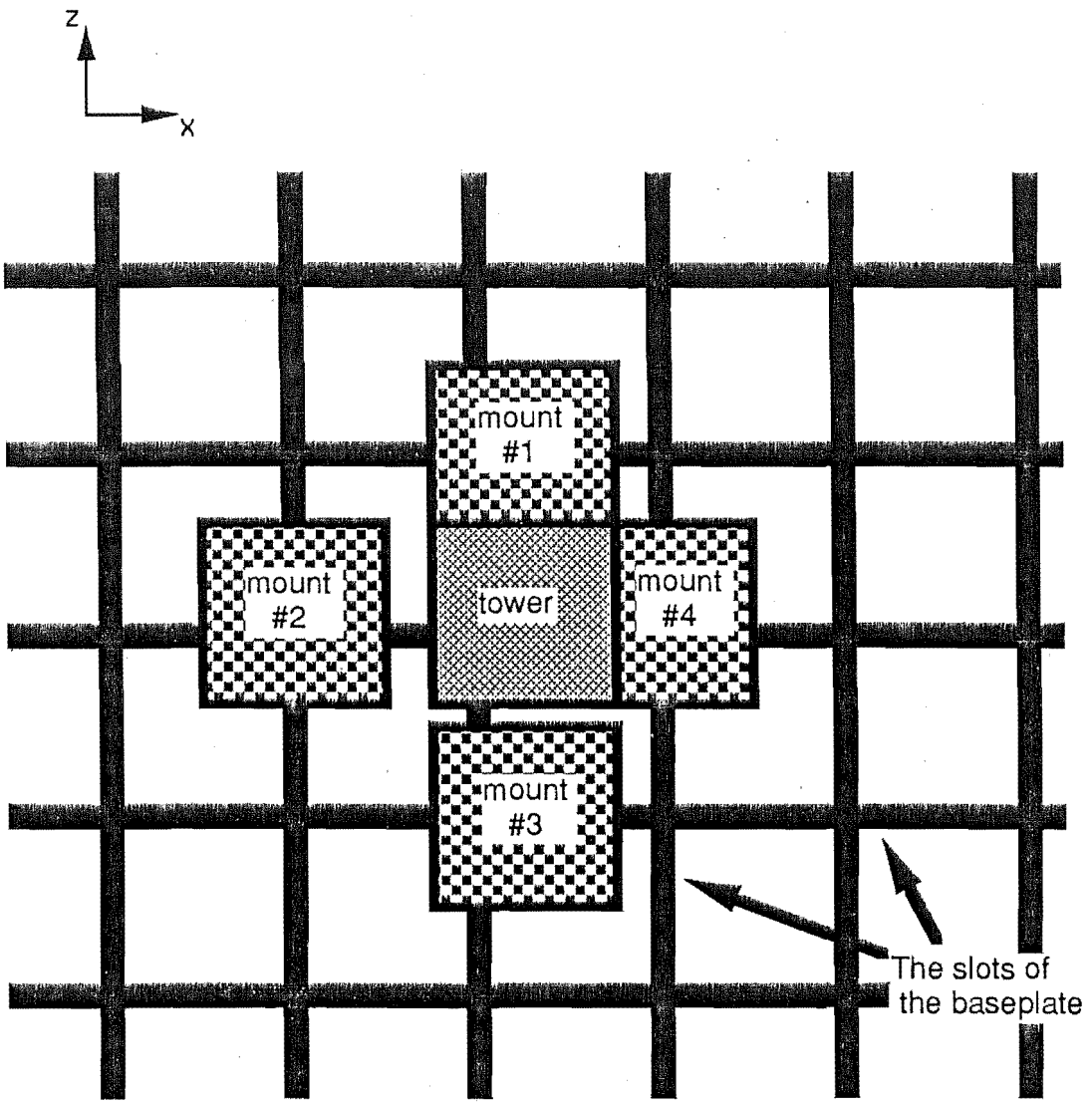


Figure 31 The four mounts of a tower

Using the position of each tower, the program automatically derives the positioning coordinates of the four mounting blocks of the tower. The distance between each mounting block and their tower are also determined and recorded. A unique spatial number is assign to each mounting block, and a matrix representation of each mounting block is generated. The process is repeated until all mounting block matrices are generated, their positions determined, and the distance from their tower recorded.

The program then constructs each mounting block, and automatically searches and identifies any intersection between the mounting block and the input, the mounting block and the towers, and the mounting block and other mounting blocks. If any intersection occurs, the program records the type of intersection and the coordinates of the intersection. Using the above information, the program determines the status of the mounting block. The process is repeated until all mounting block statuses are determined.

Mounting block and tower status

There are six types of mounting block status, namely, clear, fail, share, alternate, connect and update.

Clear status is given to a mounting block that is free of intersection.

Fail status is given to a mounting block that has interfered with another object and the intersection cannot be rectified.

Share status is given to a mounting block that intersects another mounting block and is in the same position as that mounting block

(when two mounting blocks coincides); such a mounting block can be shared by two towers (figure 32).

Alternate status is given to a mounting block that intersects another mounting block and is not in the same position as that mounting block (figure 33).

Connect status is given to a mounting block that intersects another tower which is aligned to its own tower (figure 34). It indicates that the two towers may be connected and combined into a single structure. Therefore the single structure can be mounted by mounting either tower.

Update status is given to a mounting block that intersects a sector of tower A , however the opposite sector of this tower must not have a mounting block that has a fail status (figure 35).

The principles governing the determination of the mounting block status are organised into a decision table and are shown in Table 14.

As mentioned above, each tower theoretically, can have four mounting blocks. The combination of the status of these four mounting blocks determines the status of the tower. Example of a tower status : 'fail-clear-share-fail'.

Selecting and assigning mounting blocks

Some towers may not have any mounting block that is clear of intersection. Therefore it is necessary to select and assign the mounting blocks to towers. This is based on a priority list of the tower statuses. Table 15 shows the priority list of the tower statuses.

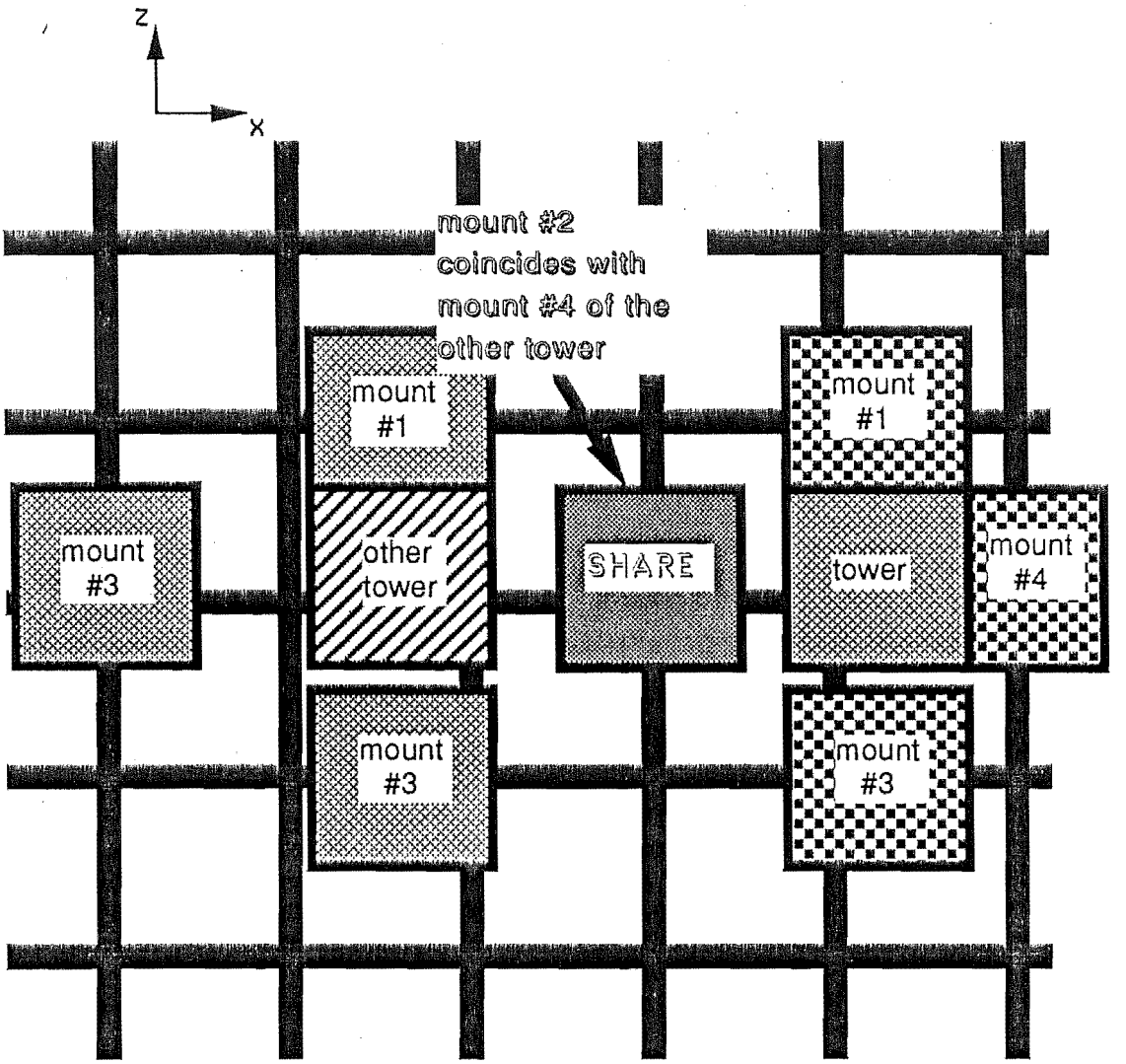


Figure 32 Share status

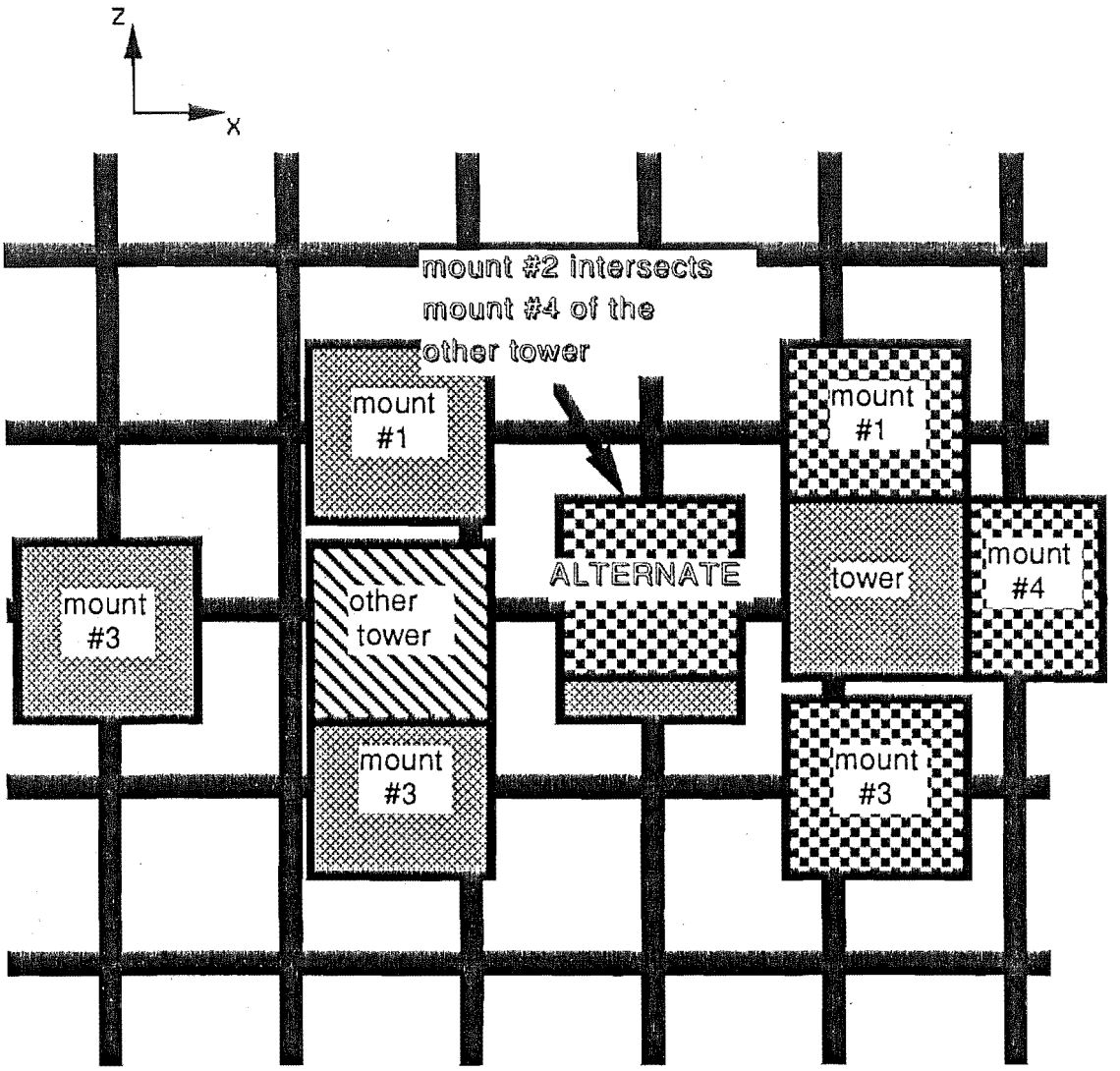


Figure 33 Alternate status

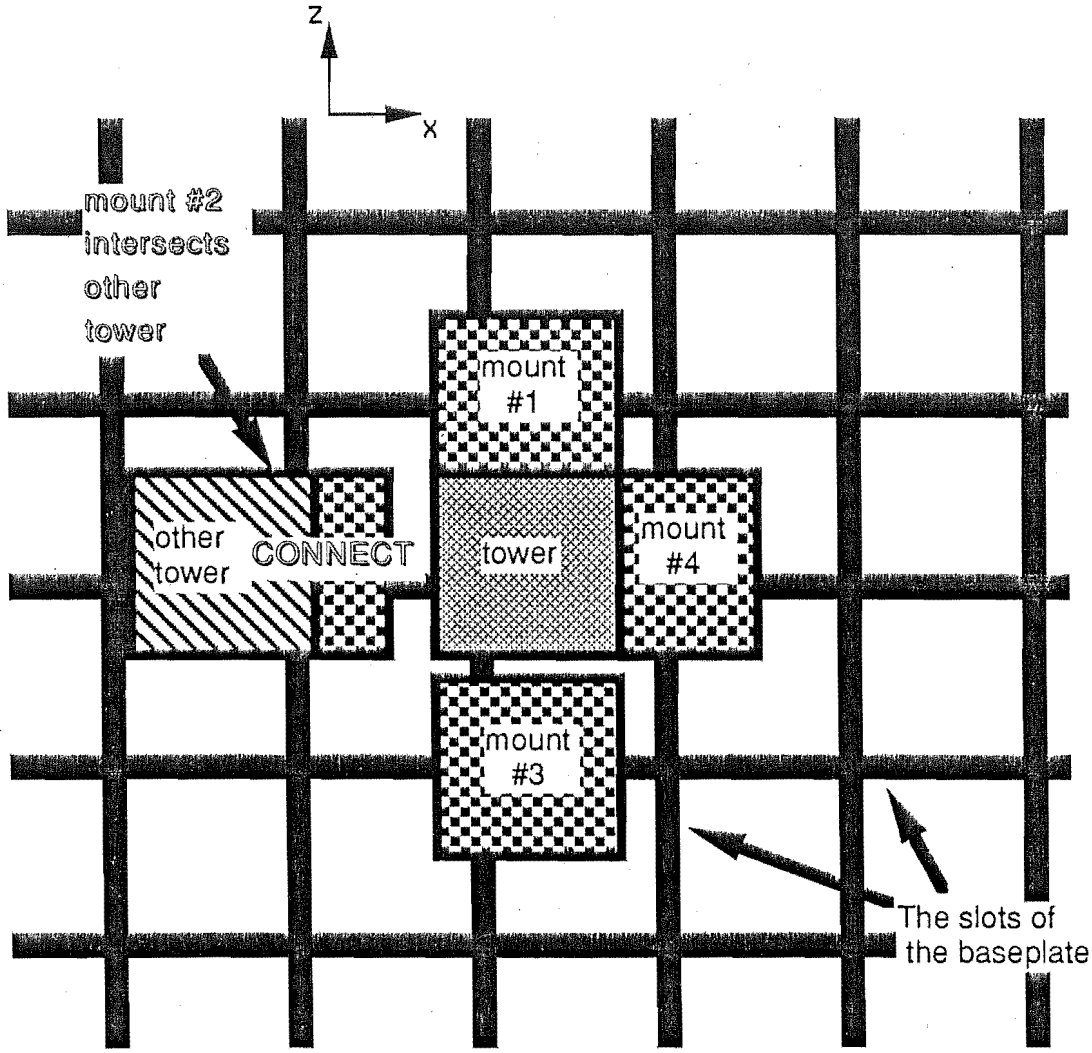


Figure 34 Connect status

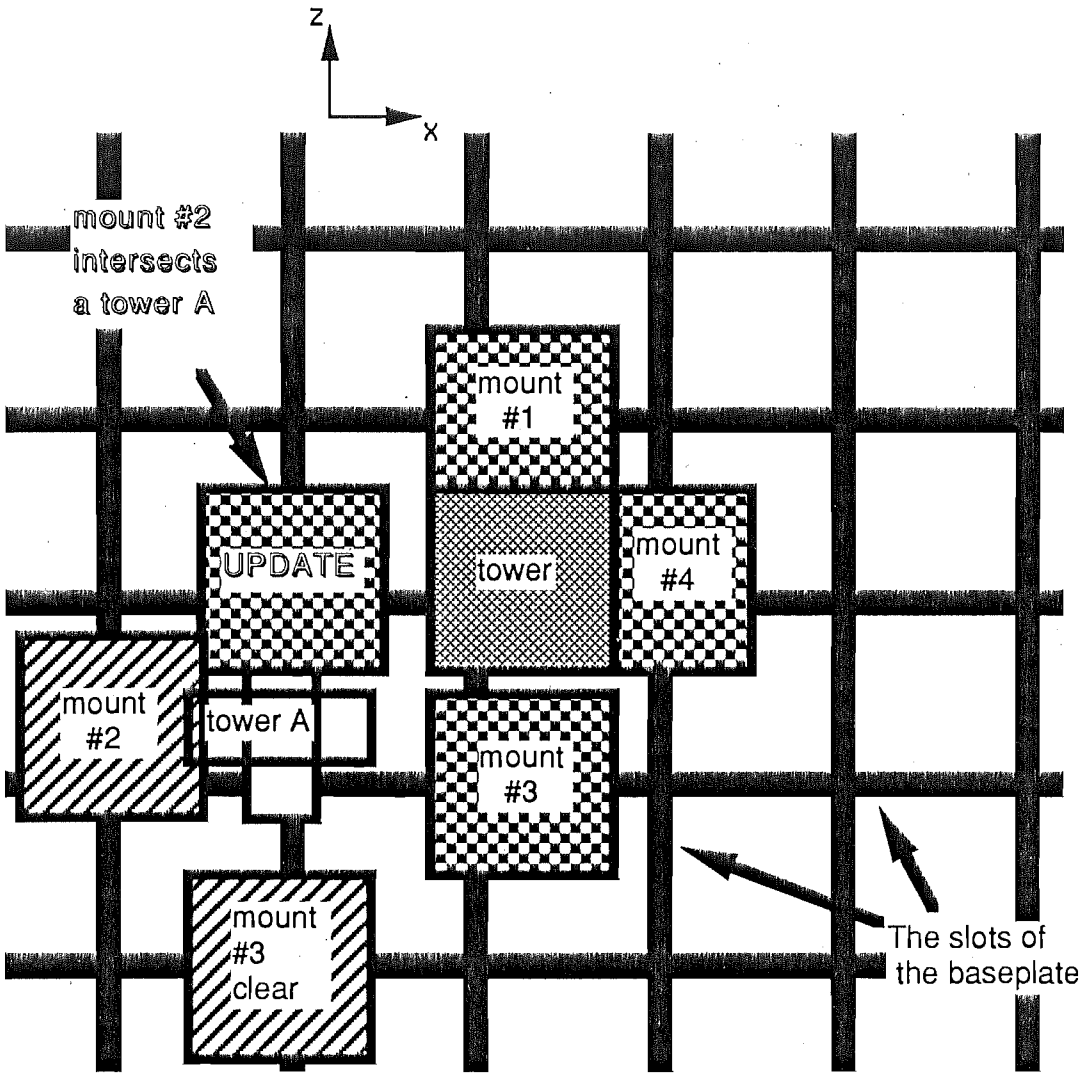


Figure 35 Update status

TABLE 14
MOUNTING BLOCK STATUS

CONDITION STUB	CONDITION ENTRIES									
solid vs mount	no	yes	no	---	no	no	---	---	no	no
mount vs mount	no	---	---	---	yes	yes	---	---	---	---
A vs mount	no	---	yes	yes	---	no	---	---	no	yes
opposite sector of A : fail ?	---	---	no	yes	---	---	---	---	---	---
B/C vs mount	no	---	no	---	---	no	yes	---	yes	no
D/Dir vs mount	no	---	no	---	no	no	---	yes	no	no
aligned with tower ?	---	---	no	no	---	---	no	no	yes	yes
coincides with mount ?	---	---	---	---	yes	no	---	---	no	no
ACTION STUB	**ACTION ENTRIES**									
clear status	X									
fail status		X		X			X	X		
update status			X							
connect status									X	X
share status					X					
alternate status						X				

ABBREVIATIONS	REPRESENTS
A	tower A
B	tower B
C	tower C
D	tower D
Dir	Direct tower

TABLE 15
PRIORITY LIST OF THE TOWER STATUS

CONDITION STUB	CONDITION ENTRIES											
3 FAIL + ?	1											
4 CLEAR		2										
3 CLEAR + ?		3										
2 CLEAR + ?		4										
1 CLEAR + ?		5										
4 SHARE			6									
3 SHARE + ?			7									
2 SHARE + ?			8									
1 SHARE + ?			9									
2 FAIL + ?				10	10	10						
1FAIL + ?							11	11	11			
0 FAIL + ?										12	12	12
any alternate / update ?	---	---	---	no	no	yes	no	no	yes	no	no	yes
Lowest level ?	---	--	---	yes	no	---	yes	no	---	yes	no	---
ACTION STUB	ACTION ENTRIES											
assign to ?	X											
assign to the nearest clear		X										
assign to the nearest share			X									
assign to the nearest alternate						1st			1st			1st
assign to the nearest update						2nd			2nd			2nd
proceed to the next level					X			X			X	
system fail				X			X			X		

When a tower is assigned to one of its mounting blocks, the existence of that mounting block is confirmed, and the other three mounting blocks belonging to the same tower has to be removed to allow maximum space for other towers to be mounted.

The above process causes some re-adjustment in the mounting block status.

Confirming the existence a mounting block will result in the failing of any mounting block that intersects it. Example of status change of other towers : alternate to fail, update to fail.

Removing a mounting block will result in the clearing of any mounting blocks that intersect it. Example of status change: alternate to clear, update to clear, connect to success.

The mounting process is repeated until every unmounted tower is assigned to a mounting block. The bridging block between the tower and the mount is constructed by activating the tower-mount bridging program.

7.2 TOWER-MOUNTING BLOCK BRIDGING PROGRAM

The tower-mounting block bridging program starts by identifying those towers that have been assigned with a mounting block. The program records the position of these towers and their mounting blocks.

Using the above information, the program derives the positioning coordinates and orientation of the bridging blocks required. A unique spatial number is assign to each bridging block and a matrix

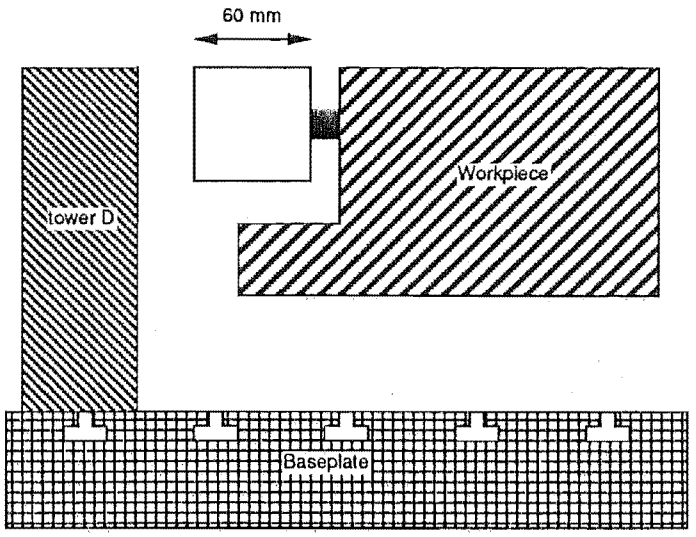
representation of each bridging block is generated. The process is repeated until all bridging block matrices are generated and their positions determined.

To minimise the number of modular blocks required, the bridging block and the bottom blocks of the tower are combined into a single block (figure 11). The length of this single block is used to determine the modular blocks required.

7.3 TOWER-LOCATOR BLOCK BRIDGING PROGRAM

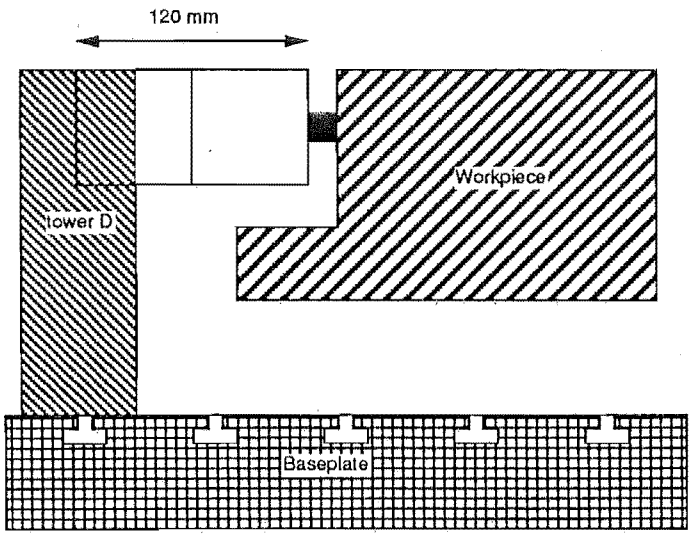
The tower-locator block bridging program starts by searching for a vertical face locator, clamp, or support. (Note that thrust locator is a special case of the vertical face locator). If no vertical face locator, clamp, or support is found the program terminates. If the vertical face locator, clamp, or support is detected, the program determines and records the coordinates of the locator, clamp, or support.

It then generates an object-matrix of the bridging block. The length of the block starts at 60 mm. The position of the bridging block matrix is derived from the position of the vertical face locator, clamp, or support. After placing the bridging block in position, the program searches for intersection between the tower and the bridging block. If no intersection is detected, then the length of the bridging block is increased by 60mm until a maximum of 180mm. The process is repeated until the bridging block intersects the tower. If intersection between tower and bridging block occurs, the program transforms the intersection space to the tower space, (figure 36) and the bridging block construction process for this locator, clamp or support is



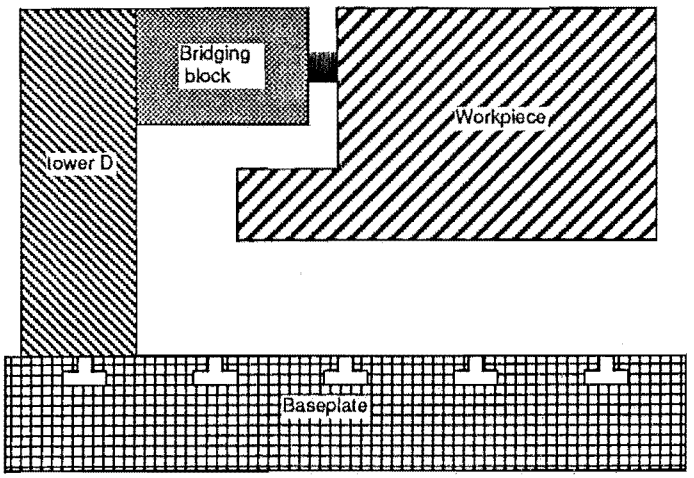
Construct bridging block of 60 mm length.

No intersection detected.



Construct bridging block of 120 mm length.

Intersection detected.



Transform intersection space into tower space.

Figure 36 Tower-locator bridging block construction

completed. The process is repeated for all vertical face locators, clamps, and supports.

CHAPTER 8

SPATIAL REPRESENTATION SYSTEM

8.1 INTRODUCTION

To develop an automatic design system of the fixture body from modular elements, a solid model representation scheme is required to describe those region of space occupied by the workpiece envelope, the machining envelope, and the fixturing elements. During the process of fixture design, the representation scheme has to identify the type of object, in some cases the specific object, and to distinguish uniquely the objects involved in an intersection.

Existing solid model representation schemes [65,66,67,68] may be classified into three main types - namely, the constructive solid geometry, the boundary representation and the cellular decomposition. In cellular decomposition [65,69,70,73,74,75,76], a solid is represented by decomposing it into cells, and representing each cell in the decomposition. Figure 37 presents the current status of the cellular decomposition representations. It is apparent that decomposition into cubical cells is the most common representation. Both octree encoding [70,74,76] and spatial enumeration [73,75] uses this representation, however their data structure is different. Generally, the octree encoding uses the hierarchical tree structure while the spatial enumeration uses list. Unlike the other workers [70,73,74,75,76] who used cubical cells, Wordenweber [72] employed

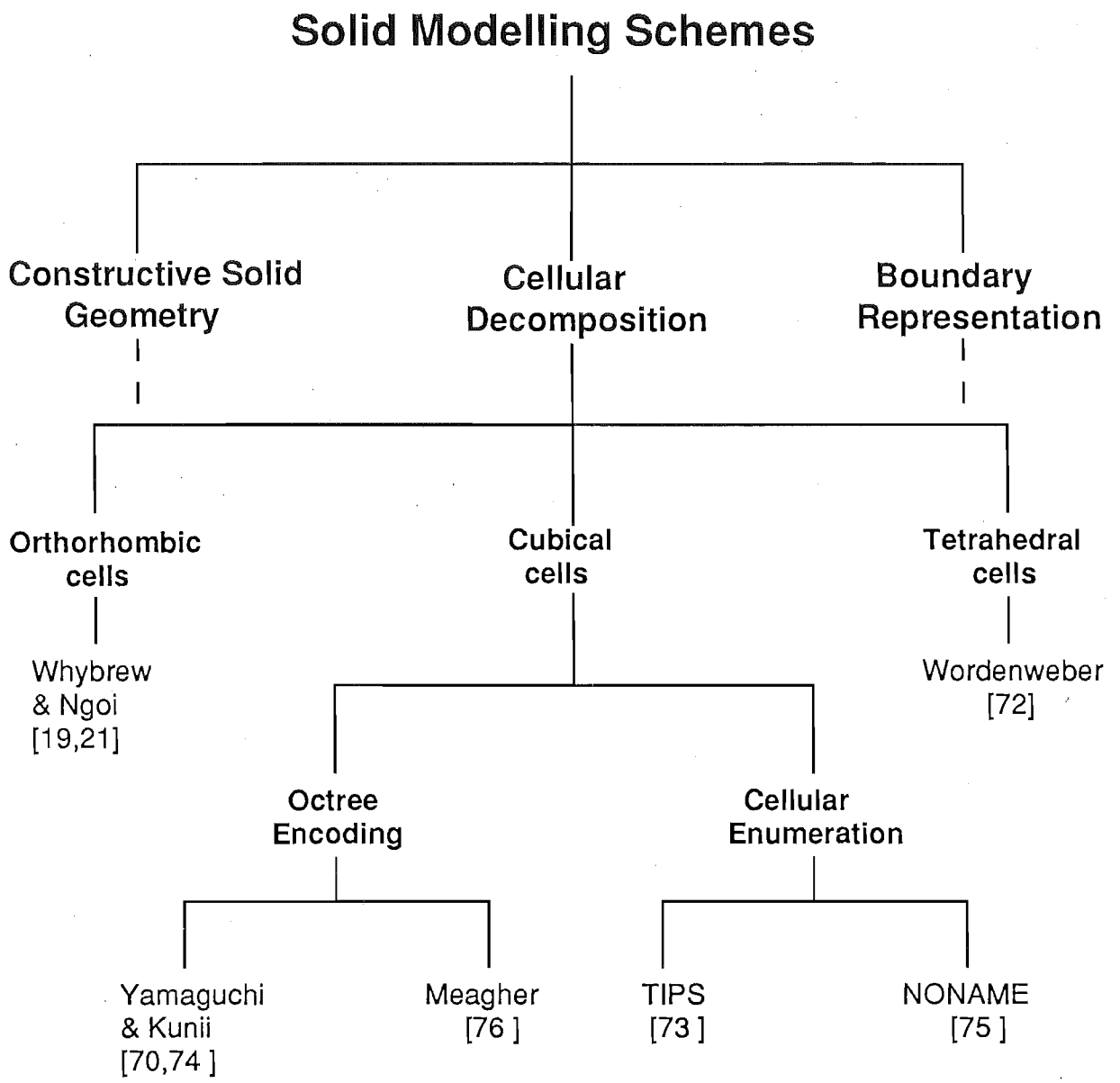


Figure 37 Current status of Cellular Decomposition Representation

tetrahedral cells in his work on automatic finite element mesh generation.

This chapter introduces a type of cellular decomposition representation. It was developed to do two things that other representation schemes will not do at all easily or quickly. Firstly, it directly provides information of which components have intersected; secondly, it is so fast that it allows checks for intersection to be made in iterative routines.

This representation scheme uses variable orthorhombic cells with a 3D matrix data structure. It has the added feature that as an integral part of the data representation, the type of space occupied by the cell can be identified.

The majority of cellular decomposition representation schemes are based on fixed size cubical cells, which provide limited resolutions. This representation scheme provides floating point definition of the cell boundaries, and the resolution is only limited by the maths processor of the computer. In practice this resolution can be reduced to decrease the demands on computing time and memory.

The orthorhombic representation is ideally suited to the shape of the modular fixturing blocks it was designed to represent. The component being held in the fixture, the tool path envelope, the locators, the supports, and the clamps may not conform to the orthorhombic cell shape, but this does not matter as they need not be represented in detail. They are inputs to the fixture body design system and only need to be represented as an envelope, which cannot be intersected

by the fixture body. The point of attachment of the locators, clamps and supports to the fixture body are on vertical or horizontal planes, and as such align with the orthorhombic cells and can be represented to whatever resolution required.

The intersection checks and other object manipulation algorithms, outlined in the section 8.4, are fast and efficient, because they are performed without recourse to complex geometrical algorithms.

This representation system is especially suitable for solving problems where the representation of the object geometry has the following specifications:

1 Exact geometric details are not important.

2 Fine resolution is only required in the direction of the principal 'x-y-z'axes.

8.2. OBJECT REPRESENTATION

8.2.1 Overall structure

A finite region, containing the entire assembly and baseplate, is represented by an array of variable orthorhombic 'cells'. Each cell may contain either a dimensional data or a geometrical data of the object. A single matrix is then used to contain both dimensional and geometrical data of the object. The orientation of the 3D matrix is defined by the x-y-z axes. The positive direction of the x axis is defined from left to right, the positive direction of the y axis is from top to bottom, and the positive direction of the z axis is going into the paper. The z axis gives depths to the 3D matrix.

The 3D matrix is described by using a string of 2 dimensional matrices. Each 2D matrix represents a 2D shape of a certain thickness. The 2D shape lies in the x-y plane and is placed at a certain z coordinate.

8.2.2 Data structure

Dimensional data

The top row of each 2D matrix, excluding the first cell, contains the coordinates x_1, x_2 etc., and the first column, excluding the first cell, contains the coordinates y_1, y_2 etc.. A series of these matrices at different depths give the third dimension. The 2D matrices at different depths are defined by the third dimension z_1, z_2 etc., which is contained in the first column of the first row of each 2D matrix.

Geometrical data

The remaining cells of each matrix are then used to represent the geometrical description of the object. The coordinates of the divisions between cells are defined as x_1, x_2 etc. in the x direction, y_1, y_2 etc. in the y direction, and z_1, z_2 etc. in the z direction. Note that the origin x_0,y_0,z_0 is defined as x=0,y=0,z=0.

The number in the 'i-j-k' location then, is a coded representation of the type of object which occupies the cell between x_{i-1} and x_i, between y_{j-1} and y_j and between z_{k-1} and z_k. The coding is such that a particular number is used to represent a particular type of object. For example, the value 30 can be used to represent one object, and the value 564 another. Empty space is represented by zero.

The relative positions of these cells of an object defines the shape of that object, and the coordinates of the object are derived from the coordinates of the cells.

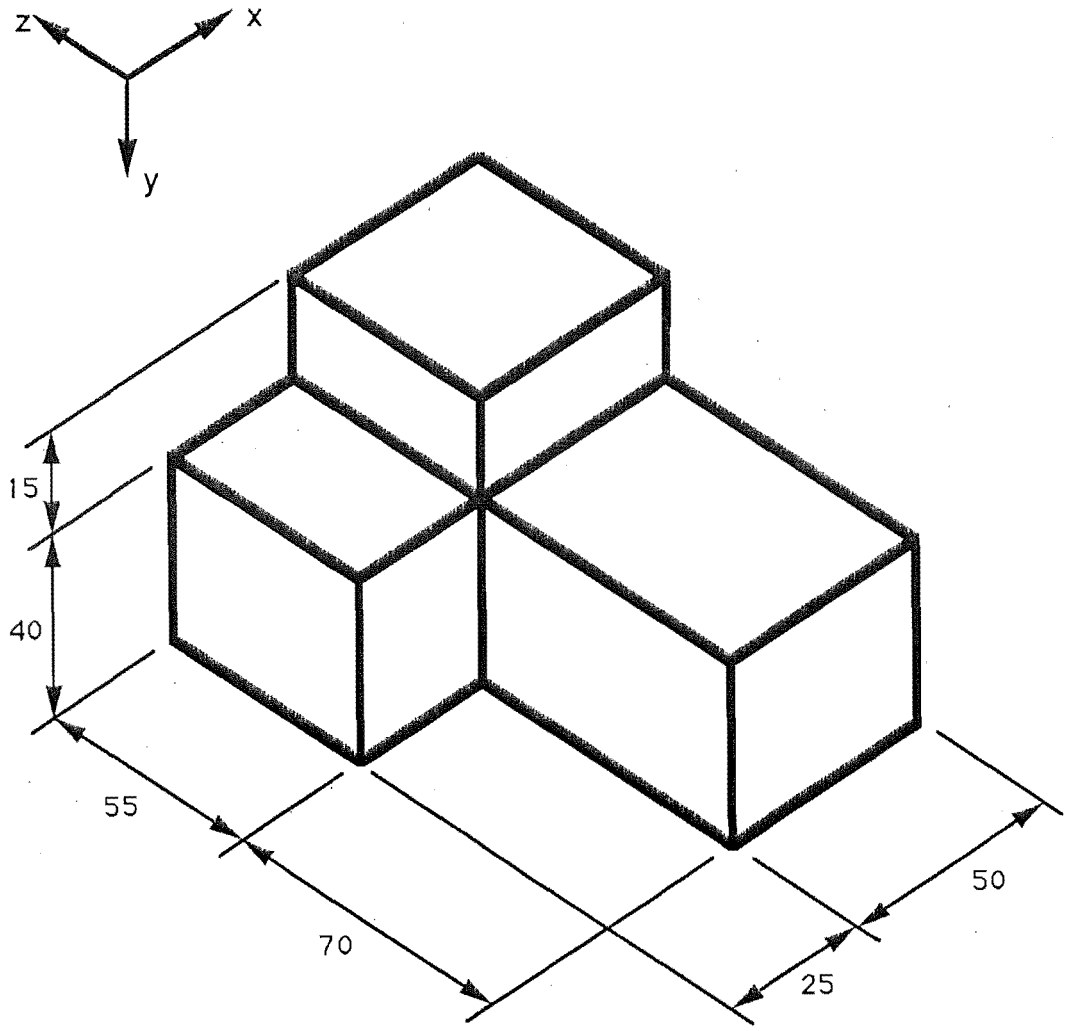
A combination of these cells form an object, and the coordinates of the object are derived from the coordinates of the cells. The relative positions of the cells of an object will determine the shape of that object. Figure 38 shows an example of an object represented using this technique.

8.2.3 An example

Let us consider the object in figure 38, to illustrate how objects are described. The 3D matrix of the object is described by using two 2D matrices. The first 2D matrix represents a rectangular block of thickness 70mm placed at z=0mm. The second 2D matrix represents a inverted 'L-shaped' block of thickness (125-70)mm placed at z=70mm.

The x coordinates of each 2D matrix are 0mm, 25mm and 75mm, and the y coordinates are 0mm, 15mm and 55mm. The first 2D matrix has a depth between 0mm and 70mm and the second between 70mm and 125mm.

The remaining cells of each matrix are then used to represent the geometrical description of the object. The number in the '2-2-2' location of the 3D matrix, 44, is a coded representation of the type of object which occupies the cell between 25mm and 75mm, between 15mm and 55mm and between 70mm and 125mm. The relative



70	25	75
15	0	0
55	0	44

125	25	75
15	0	44
55	44	44

Space matrices

Figure 38 Spatial representation of a 3-D object: an example

positions of cells containing number 44 defines the shape of that object.

8.3 SPECIAL NUMBERING SYSTEM

During the process of fixture design, the intersection of objects is represented by adding the numbers representing the intersecting objects. Since the system has to distinguish uniquely the objects involved in the intersection, these objects have to be represented by numbers belonging to a family of numbers, where the sum of any two numbers is unique.

A program is specially written to generate a set of numbers so that the sum of n numbers is always unique, where n is an integer greater than 1. Figure 39 show a set of numbers where n=2. The special system of numbers is used to represent objects. When two objects intersect, their numbers are added at the region of intersection. The system of numbers ensures that the sum is unique, therefore enabling the program to detect and identify intersecting objects. For example in figure 43, the number 74 indicates that the intersecting objects are 30 and 44.

When an intersection is detected, the main program has to refer to the rule-base for the actions to be taken.

8.4 MANIPULATION OF AN OBJECT

Besides identifying the type of object and distinguishing uniquely the objects involved in an intersection, the fixture design system requires

Numbers used to represent various spaces associated with towers	
1	1895
3	2028
7	2253
12	2378
20	2509
30	2779
44	2924
65	3154
80	3353
96	3590
122	3796
147	3997
181	4296
203	4432
251	4778
289	4850
360	5122
400	5242
474	5297
564	5750
592	5997
661	6373
774	6800
821	6924
915	7459
969	7546
1015	7788
1158	8219
1311	8502
1394	8729
1522	8941
1571	9881
1820	

Numbers used to represent various spaces associated with mounting blocks	
10199	13393
10586	14046
10897	14533
11288	14900
11613	15165
11875	15687
12033	15971
12930	

Numbers used to represent workpiece, machining envelope, locators, clamps and supports	
32619	32626
32620	32627
32621	32628
32622	32629
32623	32630
32624	32631
32625	32632

Figure 39 Number system

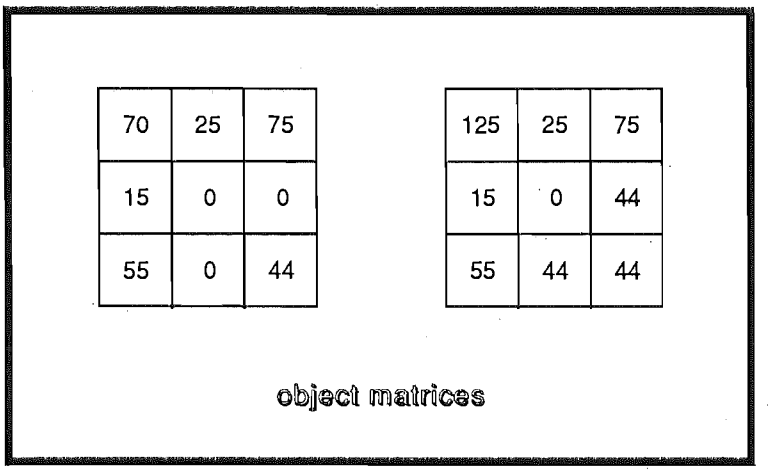
the representation system to manipulate the fixture components during the fixture design process. Example of such manipulations are:

1. Adding a new component to the existing structure.
2. Moving a fixture component to clear an interference.
3. Removing a redundant fixture component.

This section focuses on the algorithm for such operations. The algorithms of important utilities and useful tools are discussed in this section, and illustrations are given where appropriate to demonstrate the concept. All these algorithms to manipulate objects have been implemented in Turbo Pascal (version 4.0) on an IBM AT computer. An example of the application of the utilities and tools to a simple program is included in section 8.4.4.

8.4.1 Placement of an object

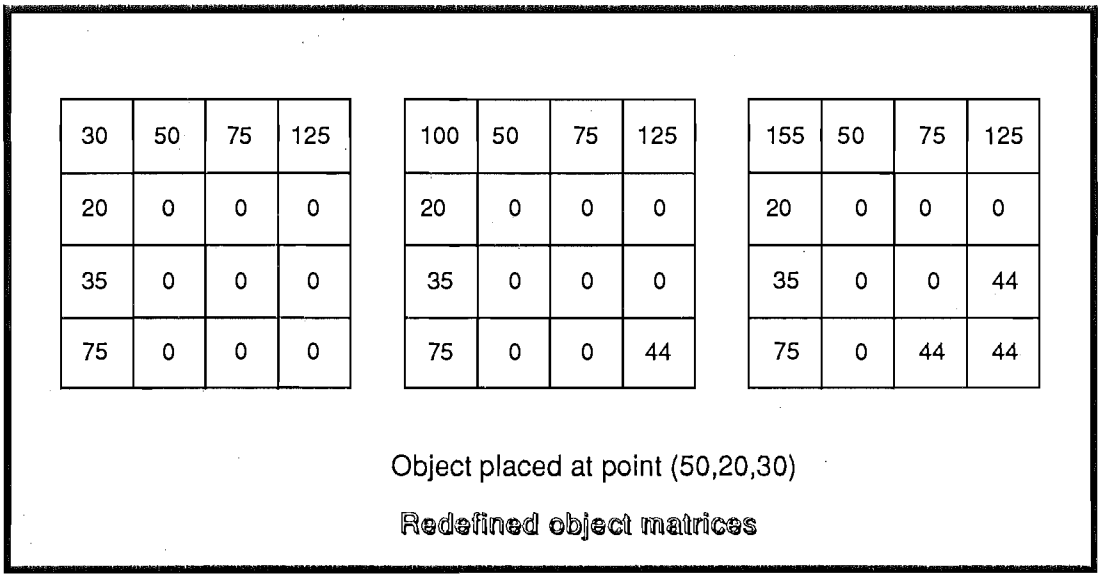
An object can be placed at any location by modifying the cell division coordinates. Figure 40 shows an example of the re-definition of a space matrix for the object discussed in the previous section. The object is placed at point (50, 20, 30). A more detail, explanation of the placement process is outlined in ALGORITHM A.



70	25	75
15	0	0
55	0	44

125	25	75
15	0	44
55	44	44

object matrices



30	50	75	125
20	0	0	0
35	0	0	0
75	0	0	0

100	50	75	125
20	0	0	0
35	0	0	0
75	0	0	44

155	50	75	125
20	0	0	0
35	0	0	44
75	0	44	44

Object placed at point (50,20,30)

Redefined object matrices

Figure 40 Placing a 3-D object at (50,20,30)

ALGORITHM A : To place an object at a specified location.

The following example illustrates the placement procedure of a simple three dimensional object.

Example: object A is to be placed at x=10,y=30,z=20.

object A

20	40	75
10	44	44
95	0	44
40	40	75
10	0	44
95	0	0

1a if the x is greater than zero insert an empty second column to the object matrix

1b if the y is greater than zero insert an empty second row to the object matrix

1c if the z is greater than zero insert an empty second depth to the object matrix

0	0	40	75
0	0	0	0
10	0	0	0
95	0	0	0
20	0	40	75
0	0	0	0
10	0	44	44
95	0	0	44
40	0	40	75
0	0	0	0
10	0	0	44
95	0	0	0

2a add x to all x coordinate if the x is greater than zero

2b add y to all y coordinate if the y is greater than zero

2c add z to all z coordinate if the z is greater than zero

0+20	0+10	40+10	75+10
0+30	0	0	0
10+30	0	0	0
95+30	0	0	0
20+20	10+10	40+10	75+10
0+30	0	0	0
10+30	0	44	44
95+30	0	0	44
40+20	10+10	40+10	75+10
0+30	0	0	0
10+30	0	0	44
95+30	0	0	0

Resultant matrix of object A placed at x=10,y=30,z=20.

20	10	50	85
30	0	0	0
40	0	0	0
125	0	0	0
40	10	50	85
30	0	0	0
40	0	44	44
125	0	0	44
60	10	50	85
30	0	0	0
40	0	0	44
125	0	0	0

8.4.2 Union of two objects

If two objects are to be added together, each matrix must first be expanded so that both have a common set of cell boundaries. The union of the two objects is then represented by adding the contents of each cell. The addition of two matrices is illustrated in figure 41 to figure 43. Figure 41 shows the two objects to be added. Figure 42 shows the expanded common matrices of the two objects and figure 43 shows the resultant matrices of the union operation. The number 74 shows where intersection has occurred. ALGORITHM B outlines the union process step by step.

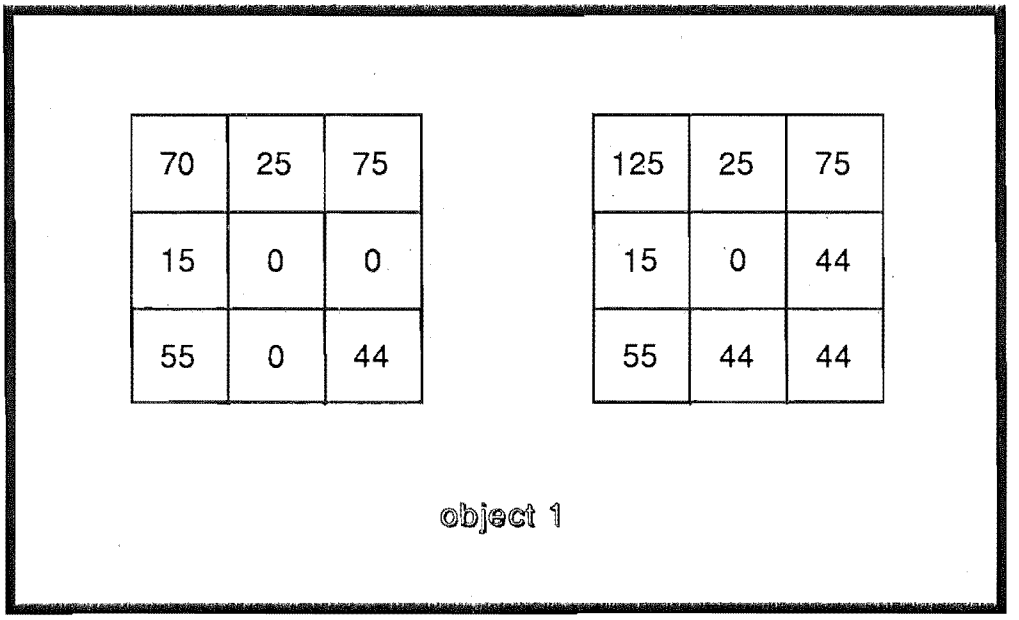


Figure 41 Union of two 3-D objects

30	25	30	75
15	0	0	0
40	0	44	44
55	0	44	44

70	25	30	75
15	0	0	0
40	0	44	44
55	0	44	44

125	25	30	75
15	0	44	44
40	0	44	44
55	44	44	44

object 1 spatial matrix expanded

30	25	30	75
15	30	30	0
40	30	30	0
55	0	0	0

70	25	30	75
15	0	0	0
40	0	0	0
55	0	0	0

125	25	30	75
15	0	0	0
40	0	0	0
55	0	0	0

object 2 spatial matrix expanded

Figure 42 Union: Common matrices

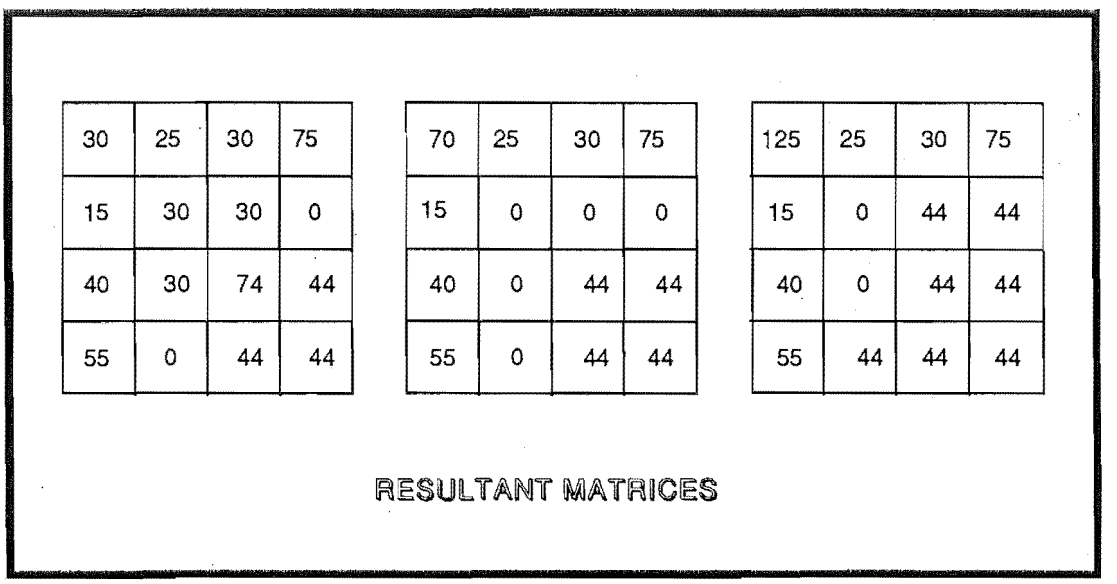


Figure 43 Union: Resultant matrices

ALGORITHM B : To union two objects.

When two object matrices are to be added, the first step is to combine their x dimensions. These dimensions are sorted in numerical order and any repetition of the dimension is removed. The procedure is repeated for the y and the z coordinates. The 'x-y-z' coordinates forms the cell boundaries (dimensions) of the new common matrix. A new expanded matrix is created for each object. At this point the content of the expanded cell is unknown, therefore it is temporarily assigned with a reserved spatial number (80).

Then the spatial information of the old object matrices is transferred to the new expanded matrix, so that the expanded matrices of both objects, which have a same common set of cell boundaries, is fully defined. By comparing the dimensions of the old and the new cell boundaries, an essential set of spaces can be transferred to the new expanded matrix. With this essential space, the program will systematically fill up the remaining space as illustrated below. The concept for two dimensional objects and three dimensional objects is the same. A two dimensional example is used to illustrate the procedure.

object A		
0	40	75
10	44	44
95	0	44

object B		
0	60	90
50	30	30
80	30	0

(1) combine the x dimensions

{ 40,75,60,90}

(2) sort and eliminate repetitions

{40,60,75,90}

(3) combine the y dimensions

{10,95,50,80}

(4) sort and eliminate repetitions

{10,50,80,95}

(5) Expansion of object A

(5a) Assign all space to reserved number 80.

0	40	60	75	90
10	80	80	80	80
50	80	80	80	80
80	80	80	80	80
95	80	80	80	80

(5b) Assign essential coordinates.

0	40	60	75	90
10	44	80	44	80
50	80	80	80	80
80	80	80	80	80
95	0	80	44	80

(5c) Operation on columns

0	40	60	75	90
10	44	80	44	80
50	0	80	44	80
80	0	80	44	80
95	0	80	44	80

(5d) Operation on rows

0	40	60	75	90
10	44	44	44	80
50	0	44	44	80
80	0	44	44	80
95	0	44	44	80

(5e) Set remaining reserved numbers to zero

0	40	60	75	90
10	44	44	44	0
50	0	44	44	0
80	0	44	44	0
95	0	44	44	0

(6) Expansion of object B

(6a) Assign all space to reserved number 80.

0	40	60	75	90
10	80	80	80	80
50	80	80	80	80
80	80	80	80	80
95	80	80	80	80

(6b) Assign essential coordinates.

0	40	60	75	90
10	80	80	80	80
50	80	30	80	30
80	80	30	80	0
95	80	80	80	80

(6c) Operation on columns

0	40	60	75	90
10	80	30	80	30
50	80	30	80	30
80	80	30	80	0
95	80	80	80	80

(6d) Operation on rows

0	40	60	75	90
10	30	30	30	30
50	30	30	30	30
80	30	30	0	0
95	80	80	80	80

(6e) Set remaining reserved numbers to zero

0	40	60	75	90
10	30	30	30	30
50	30	30	30	30
80	30	30	0	0
95	0	0	0	0

(7) The union of the two objects is then represented by adding the contents of each cell.

object A

0	40	60	75	90
10	44	44	44	0
50	0	44	44	0
80	0	44	44	0
95	0	44	44	0

object B

0	40	60	75	90
10	30	30	30	30
50	30	30	30	30
80	30	30	0	0
95	0	0	0	0

object A + object B

0	40	60	75	90
10	44+30	44+30	44+30	0+30
50	0+30	44+30	44+30	0+30
80	0+30	44+30	44+0	0+0
95	0+0	44+0	44+0	0+0

Resultant

0	40	60	75	90
10	74	74	74	30
50	30	74	74	30
80	30	74	44	0
95	0	44	44	0

8.4.3 Algorithm for other tools

Besides adding and placing objects there are other tools needed for efficient functioning of the program. To understand the algorithm, it is necessary to note that the object number refers to the unique spatial number assigned to an object, and matrix [i,j,k] refers a cell belonging to the spatial matrix representing the inputs and the fixture elements.

ALGORITHM C : Searching for an object.

Searching for an object involves comparing every cell of the matrix with the object number until a match is found.

example of application

Searching for a locator during the tower construction process.

procedure

```
for k:= 2 to depth
for j:= 2 to row
for i:= 2 to column
begin
if matrix[i,j,k] = object number
then begin
xx:=i;          yy:=j;          zz:=k;
end;
end;            (:= means assigned)
```

result

object found at column xx, row yy and depth zz

ALGORITHM D : Determine the position of an object.

example of application

To determine the locator position in order to derive the position to construct the tower.

procedure

```
for k:= 2 to depth
for j:= 2 to row
for i:= 2 to column
begin
if  matrix[i,j,k] = object number and
matrix[i-1,j,k] < > object number and
matrix[i,j-1,k] < > object number and
matrix[i,j,k-1] < > object number
then begin
xx:=i;          yy:=j;          zz:=k;
end;
end;
```

(:= means assigned)

result

x coordinate of the object = matrix[xx-1,1,1].

y coordinate of the object = matrix[1,yy-1,1].

z coordinate of the object = matrix[1,1,zz-1].

ALGORITHM E : Determine the position of an intersection.

example of application

To determine the position of an intersection so that the program could either reposition the towers or replace the towers by a combination tower.

procedure

```
for k:= 2 to depth
for j:= 2 to row
for i:= 2 to column
begin
if matrix[i,j,k] = (object number 1 + object number 2) and
matrix[i-1,j,k] <> (object number 1 + object number 2) and
matrix[i,j-1,k] <> (object number 1 + object number 2) and
matrix[i,j,k-1] <> (object number 1 + object number 2)
then begin
xx:=i;          yy:=j;          zz:=k;
end;
end;
```

result

x coordinate of object 1 + object 2 = matrix[xx-1,1,1].

y coordinate of object 1 + object 2 = matrix[1,yy-1,1].

z coordinate of object 1 + object 2 = matrix[1,1,zz-1].

Note : object number 1 represents object number of object 1 and object number 2 represents object number of object 2.

ALGORITHM F : Search for an intersection and identification of the objects involved in the intersection.

example of application

To determine the type of an intersection so that the program could decide the course of action.

procedure and result

if matrix[i,j,k] = (object number 1 + object number 2)

then objects involved in the intersection

are object 1 and object 2

ALGORITHM G : Transform one object into another.

example of application

To transform an indirect tower into a direct tower.

procedure

```
for k:= 2 to depth
for j:= 2 to row
for i:= 2 to column
begin
if matrix[i,j,k] = object number 1
then matrix[i,j,k] := object number 2
end;
```

(:= means assigned)

ALGORITHM H : Removal of an unwanted object.

example of application

To remove existing towers before constructing new combination tower.

procedure

```
for k:= 2 to depth
for j:= 2 to row
for i:= 2 to column
begin
if matrix[i,j,k] = object number
then matrix[i,j,k] := 0
end;
```

(:= means assigned)

8.4.4 Example of application of the tools and utilities.

Tower-locator bridging program

1 **Search** for a vertical face locator, clamp, or support (Note that thrust locator is a special case of the vertical face locator)

2 If no vertical face locator, clamp, or support is detected, then go to step 14.

3 **Determine the position** of the vertical face locator, clamp, or support detected.

4 Initiate n to 1

5 Set length of bridging block to n X 60 mm.

6 Generate a object-matrix of the bridging blocks.

7 **Derive the position** of the bridging block matrix from the position of the vertical face locator, clamp, or support.

8 **Place** the bridging block in position.

9 **Union** the bridging block to the assembly.

10 **Search for intersection** between the tower and the vertical face locator, clamp, or support.

11 If no intersection is detected, then assign n to n+1 and repeat step 4 to 9 until n>3.

12 If intersection between tower and bridging block is detected, then **transform** intersection to tower.

13 Reactivate the program (step 1 to step 12)

14 End of program

CHAPTER 9

SYSTEM PROGRAM

9.1 THE SYSTEM RESTRICTION

9.1.1 Modular programming and dynamic variable

There are two major problems :

1 The programs need large code segments.

Since the size of a single code segment cannot exceed 64K, modular programming concept is required. The large code segment is divided into smaller units linked together by a main program.

2 The programs need large data segments to contain the three dimensional matrices.

Since the conventional data segment is fixed in size and can only contain between 16K to 64K of data, it is not acceptable . Therefore dynamic variables, which utilises the heap segment, are used.

9.1.2 Accuracy versus complexity

Turbo pascal 4.0 supports a few integer data types, each of which has a different range of values. Examples of the integer data types are words, long integers and short integers. The integer data type used for this program is words.

The memory requirement for the program to contain the spatial-matrices is calculated as follows:

10 matrices is required during the design process.
Each word requires 2 bytes (integers ranging from 0 to 65535).

Each matrix size = 30 columns X 30 rows X 30 depths.
= 27,000 cells
= 27,000 words
= 54,000 bytes

(maximum size of a single variable is 65,521 bytes)

10 matrices = 270,000 cells
= 270,000 words
= 540,000 bytes

(The maximum size of the heap is 655,360 bytes)

The maximum total memory available using the DOS is 640K (655,360 bytes). Therefore it is apparent that the memory restriction is a problem. However with a matrix size of 30 by 30 by 30, objects of reasonable complexity can be represented.

Dynamic variables must be used, since only dynamic variables have access to the heap.

Using word requires only 2 bytes for each cell, however the numbers used to represent the objects and object dimensions, are restricted to integers ranging from 0 to 65535. If higher accuracy is required, long integers or even real numbers may be used. If long integer is used, it allows integers between -2147483648 to 2147483647. Each long

integer requires 4 bytes per cell, and therefore uses twice as much memory. Therefore a compromise between accuracy and complexity is required, unless the amount of memory available can be increased.

9.2 THE PROGRAM

9.2.1 JEN3D.PAS

This program generates all towers required for the fixture construction process. It is too large and has to be subdivided into smaller modules. They are :

(a) JEN3D1.PAS:

This program generates the horizontal face general tower, the vertical face general tower (north), the vertical face general tower (south), the vertical face general tower (east), and the vertical face general tower (west)

(b) JEN3D2.PAS:

This program generates the thrust tower (north), and the thrust tower south.

(c) JEN3D3.PAS:

This program generates the thrust tower (east), and the thrust tower (west).

(d) JEN3D4.PAS:

This program generates the horizontal special tower, and the input file for the tower mounting program.

The main program together with its modules are called the tower generation program. The main program automatically compiles and links the subprograms with the utilities and rulebases to create an executable file (JEN3D.EXE).

9.2.2 JAY3D.PAS

This program mounts all towers to the baseplate. It is a big program and has to be subdivided into smaller modules. They are :

(a) JAY3D1.PAS:

This program mounts the thrust tower to the baseplate and generates the advantageous coordinates for the 'workpiece unit' with respect to the baseplate.

(b) JAY3D2.PAS:

This program tests each advantageous coordinate and selects the position with the maximum number of direct mountings.

(c) JAY3D3.PAS:

This program places the workpiece unit at the selected position and mounts all direct towers. It also mounts any other tower that is aligned to the direct towers.

(d) JAY3D4.PAS:

This program generates the mounting blocks. It also determines the mounting block status and tower status.

(e) JAY3D5.PAS:

This program assigns the mounting block to towers.

(d) JAY3D6.PAS:

This program generates the bridging blocks between the towers and the mounts.

The main program together with its modules are called the tower mounting program. The main program automatically compiles and links the subprograms with the utilities to create an executable file (JAY3D.EXE).

9.2.3 KAB3D.PAS

This program generates bridging blocks between the tower and the vertical face locators, clamps, and supports (includes thrust locators). The modules that belongs to this program are : KAB3D1.PAS, KAB3D2.PAS, SHIMCS, and SHIMDS. The main program automatically compiles and links the subprograms with the utilities to create an executable file (KAB3D.EXE).

9.2.4 LIM3D.PAS

This program generates the final output and is divided into smaller modules. They are :

(a) LIM3D1.PAS:

This program generates a part list of fixture elements - the type of block, the orientation and the position.

(b) LIM3D2.PAS:

This program generates the parametric program for I/EMS graphic station.

The main program automatically compiles and links the subprograms with the utilities to create an executable file (LIM3D.EXE).

9.2.5 FIXTURE.BAT

The above programs, JEN3D.EXE, JAY3D.EXE, KAB3D.EXE and LIM3D.EXE, are executed one after the other in strict order, and require no interaction from the user. Therefore they can be linked together by a batch file and can be run unattended. The batch file (FIXTURE.BAT) consists of four consecutive commands, each of which runs only one program. This facility allows the user to run only one program namely, FIXTURE.BAT, instead of four separate programs.

9.2.6 RULEBASES

The tower generation program is dependent on the rulebases to provide the knowledge to resolve any intersection which may occur during the tower construction process.

The rulebases are a series of subprograms that store the rules governing the actions to be taken when a particular intersection is detected and identified. There are seven rulebases, namely, U3DTA.PAS, U3D1TA.PAS, U3D2TA.PAS, U3DTC.PAS, U3DTCS.PAS, U3DTCE.PAS, and U3DTCW.PAS.

The first three rulebases store the rules governing intersections relating to the horizontal face towers, while the latter four rulebases store rules governing intersections relating to the vertical face towers.

9.2.7 UTILITIES

The utilities programs form the backbone of system programs. They are required by the system program to manipulate objects and to monitor progress (Refer to chapter 8). The three main utility programs are :

U3DUNION.PAS:

This program combines two spatial matrices into a resultant common matrix containing both entities. Refer to section 8.1.3 and appendix A for more detail.

U3DPV.PAS:

This program contains the utility program to place object matrices at a specified point. Refer to section 8.1.4 and appendix B for more detail. It also contains other few useful tools to manipulate objects. Refer to appendix C for more detail.

U3DPICT.PAS:

This program converts the matrix representation of the object into an orthogonal projection and displays three views of the object on the screen. This is very useful during the fixture construction process to monitor the process.

CHAPTER 10

DESIGN EXAMPLES

10.1 A DETAIL DESIGN EXAMPLE

Figure 44 shows the input to the design system, the component and the fixture elements. The workpiece is shown in red, and the locators and supports are shown in blue, and the clamps are shown in green.

A matrix spatial representation of this input, as shown in figure 45, is generated and transmitted to the tower generation program. (This conversion is presently done manually but is soon to be made automatic).

In figures 46,47,48 and 52, the locators, clamps, and supports are represented in blue, while the baseplate is shown in green.

In figure 46, the horizontal face towers have been generated and positioned. In figure 47, the vertical face towers have been generated and positioned. The workpiece is removed, in these figures, for clarity. The upper portion of each tower is represented by a yellow box, and the lower portion is represented by a red box. The towers are represented by boxes instead of the actual modular blocks at this stage, because the orientation of the modular blocks that forms these towers are still unknown.

Figure 48 shows the final output. The workpiece is again removed for clarity. All towers, mounting blocks, and bridging blocks are

represented in yellow. A matrix spatial representation of this output is shown in figure 49.

The baseplate has been automatically selected. All the towers are mounted on the baseplate, and this information has been used to generate a parametric program as shown in figure 50. It selects and positions graphic cells for display using the Intergraph Engineering Modelling System (I/EMS). A listing of the fixture elements, together with their position and orientation, is also available (figure 51).

In Figure 52, the output and input graphics have been combined to show the component, represented in red, in place in the fixture.

10.2 OTHER DESIGN EXAMPLES

Figures 53 to 62 show other examples of fixtures generated automatically using the system.

Figure 53 shows a simple rectangular block fixtured by six locators and two clamps using the "3-2-1" locating principle. In this example all towers holding the locators and clamps are directly mounted onto the slots of the baseplate.

In figure 54, there are six towers holding seven locators. Therefore one of the towers is holding two locators and this tower is called a combination tower. The combination tower is constructed to replace two intersecting towers (one tower B and one tower C). The rules governing such replacement is contained in table 9.

The clamp shown in figure 54 is held by an indirect tower.

Figure 55 illustrates a combination tower holding a thrust locator and a vertical face locator. Since the two locators are aligned and parallel, and their towers intersect, the towers are combined in accordance to the rules in table 5.

Figure 56 illustrates a combination tower holding two vertical face locators. The combination tower is constructed to replace two intersecting tower Cs. The intersection is illustrated in figure 23 and the rules governing this intersection is contained in table 6.

Figures 57,58,59,60,61 and 62 show that the system can assemble modular fixtures to hold workpieces of different configurations.

Figures 54,55 and 56 have demonstrated the working of the rulebase, while the other examples have demonstrated the general working of the system.

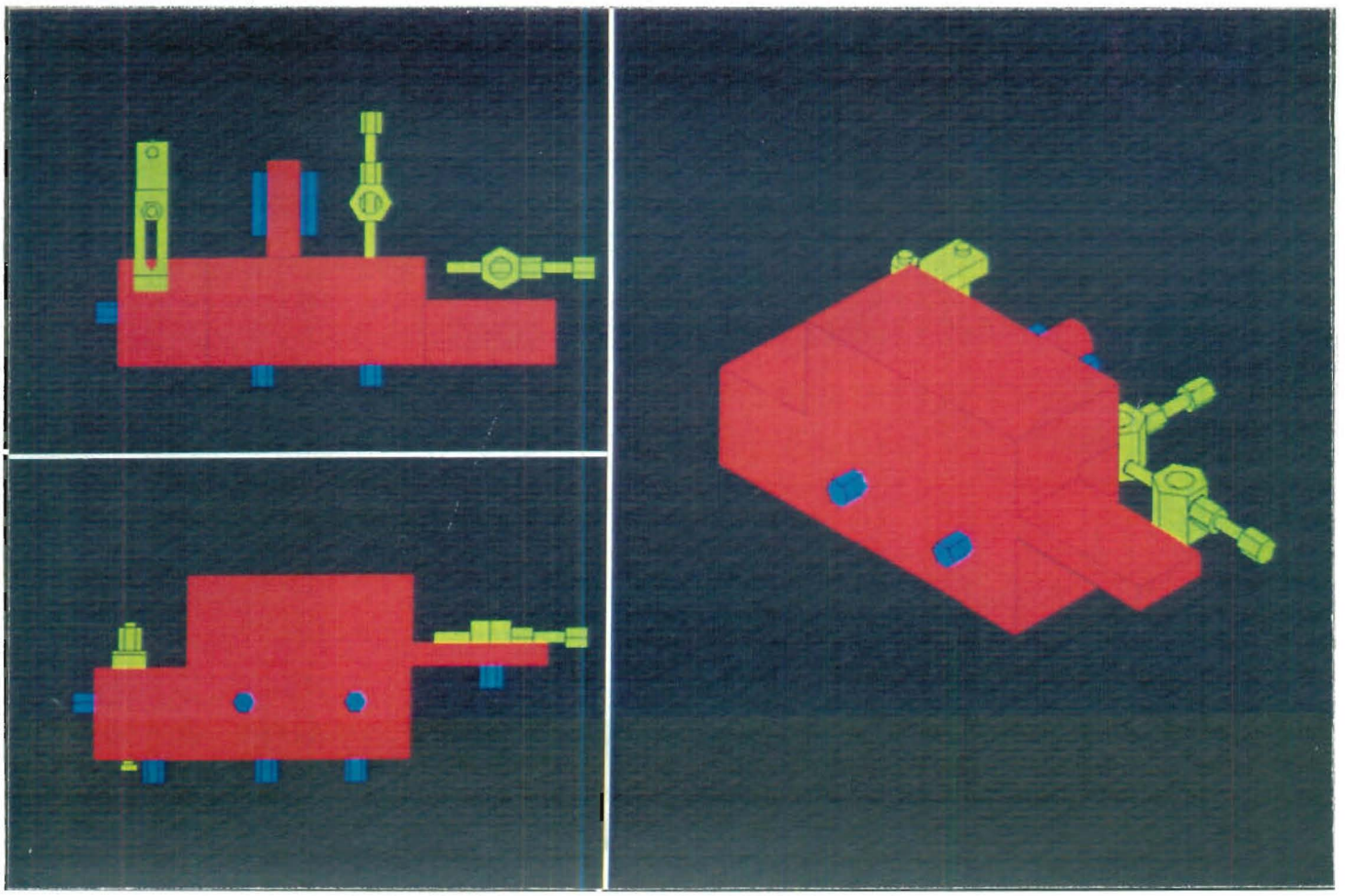


Figure 44 Inputs

MATRIX SPATIAL REPRESENTATION OF THE INPUT

20	20	60	80	100	140	160	180	200	240	260	300	360	380	400
60	0	0	0	0	0	0	0	0	0	0	0	0	0	0
80	0	0	0	0	0	0	0	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	0	0	0	0	0	0
120	0	0	0	0	0	626	0	0	0	626	0	0	0	0
160	0	0	0	0	0	0	0	0	0	0	0	0	0	0
180	0	0	0	0	0	0	0	0	0	0	0	0	0	0
240	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40	20	60	80	100	140	160	180	200	240	260	300	360	380	400
60	0	0	0	0	632	632	632	632	632	632	632	0	0	0
80	0	0	0	0	632	632	632	632	632	632	632	632	632	632
100	0	632	632	632	632	632	632	632	632	632	632	0	0	0
120	0	632	632	632	632	632	632	632	632	632	632	0	0	0
160	0	632	632	632	632	632	632	632	632	632	632	0	0	0
180	0	0	0	0	0	0	0	0	0	0	0	0	0	0
240	0	0	0	0	0	0	0	0	0	0	0	0	0	0

60	20	60	80	100	140	160	180	200	240	260	300	360	380	400
60	0	0	0	0	632	632	632	632	632	632	632	0	0	0
80	0	0	0	0	632	632	632	632	632	632	632	632	632	632
100	0	632	632	632	632	632	632	632	632	632	632	0	621	0
120	0	632	632	632	632	632	632	632	632	632	632	0	0	0
160	0	632	632	632	632	632	632	632	632	632	632	0	0	0
180	0	0	0	0	0	0	0	0	0	621	0	0	0	0
240	0	0	0	0	0	0	0	0	0	0	0	0	0	0

80	20	60	80	100	140	160	180	200	240	260	300	360	380	400
60	0	0	0	0	632	632	632	632	632	632	632	0	0	0
80	0	0	632	0	632	632	632	632	632	632	632	632	632	632
100	0	632	632	632	632	632	632	632	632	632	632	0	0	0
120	628	632	632	632	632	632	632	632	632	632	632	0	0	0
160	0	632	632	632	632	632	632	632	632	632	632	0	0	0
180	0	0	621	0	0	0	0	0	0	0	0	0	0	0
240	0	0	0	0	0	0	0	0	0	0	0	0	0	0

100	20	60	80	100	140	160	180	200	240	260	300	360	380	400
60	0	0	0	0	632	632	632	632	632	632	632	0	0	0
80	0	0	632	0	632	632	632	632	632	632	632	0	0	0
100	0	632	632	632	632	632	632	632	632	632	632	0	0	0
120	0	632	632	632	632	632	632	632	632	632	632	0	0	0
160	0	632	632	632	632	632	632	632	632	632	632	0	0	0
180	0	0	0	0	0	0	621	0	0	0	0	0	0	0
240	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45(1)

120	20	60	80	100	140	160	180	200	240	260	300	360	380	400
60	0	0	0	0	632	632	632	632	632	632	632	0	0	0
80	0	0	632	0	632	632	632	632	632	632	632	632	621	632
100	0	632	632	632	632	632	632	632	632	632	632	0	0	0
120	0	632	632	632	632	632	632	632	632	632	632	0	0	0
160	0	632	632	632	632	632	632	632	632	632	632	0	0	0
180	0	0	0	0	0	0	0	0	0	0	0	0	0	0
240	0	0	0	0	0	0	0	0	0	0	0	0	0	0

160	20	60	80	100	140	160	180	200	240	260	300	360	380	400
60	0	0	0	0	0	632	632	632	0	0	0	0	0	0
80	0	0	632	0	0	632	632	632	0	0	0	0	0	0
100	0	0	632	0	0	632	632	632	0	632	0	0	0	0
120	0	0	632	0	0	632	632	632	0	0	0	0	0	0
160	0	0	632	0	0	0	0	0	0	0	0	0	0	0
180	0	0	632	0	0	0	0	0	0	0	0	0	0	0
240	0	0	632	0	0	0	0	0	0	0	0	0	0	0

180	20	60	80	100	140	160	180	200	240	260	300	360	380	400
60	0	0	0	0	0	632	632	632	0	0	0	0	0	0
80	0	0	632	0	0	632	632	632	0	0	0	0	0	0
100	0	0	632	0	0	632	632	632	0	632	0	0	0	0
120	0	0	632	0	0	632	621	632	0	621	0	0	0	0
160	0	0	632	0	0	0	0	0	0	0	0	0	0	0
180	0	0	632	0	0	0	0	0	0	0	0	0	0	0
240	0	0	632	0	0	0	0	0	0	0	0	0	0	0

200	20	60	80	100	140	160	180	200	240	260	300	360	380	400
60	0	0	0	0	0	632	632	632	0	0	0	0	0	0
80	0	0	632	0	0	632	632	632	0	0	0	0	0	0
100	0	0	632	0	0	632	632	632	0	632	0	0	0	0
120	0	0	632	0	0	632	632	632	0	0	0	0	0	0
160	0	0	632	0	0	0	0	0	0	0	0	0	0	0
180	0	0	632	0	0	0	0	0	0	0	0	0	0	0
240	0	0	632	0	0	0	0	0	0	0	0	0	0	0

To show the whole matrix clearly, only the last 3 digits of the spatial numbers are displayed.

solid space (32632)	632
horizontal face locator (32621)	621
thrust locator - north (32626)	626
thrust locator - east (32628)	628

Figure 45(2)

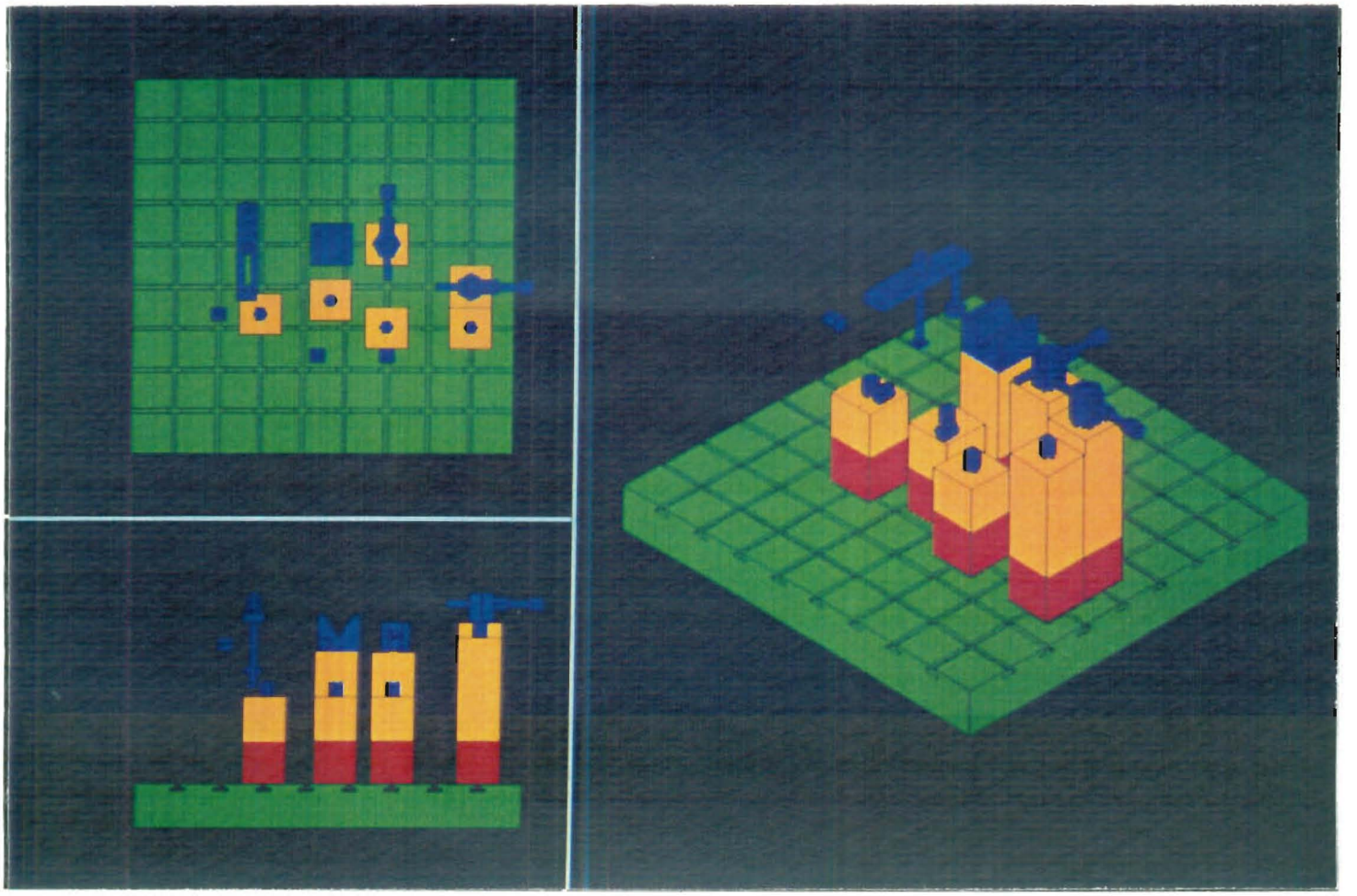


Figure 46 Horizontal face towers added (workpiece removed)

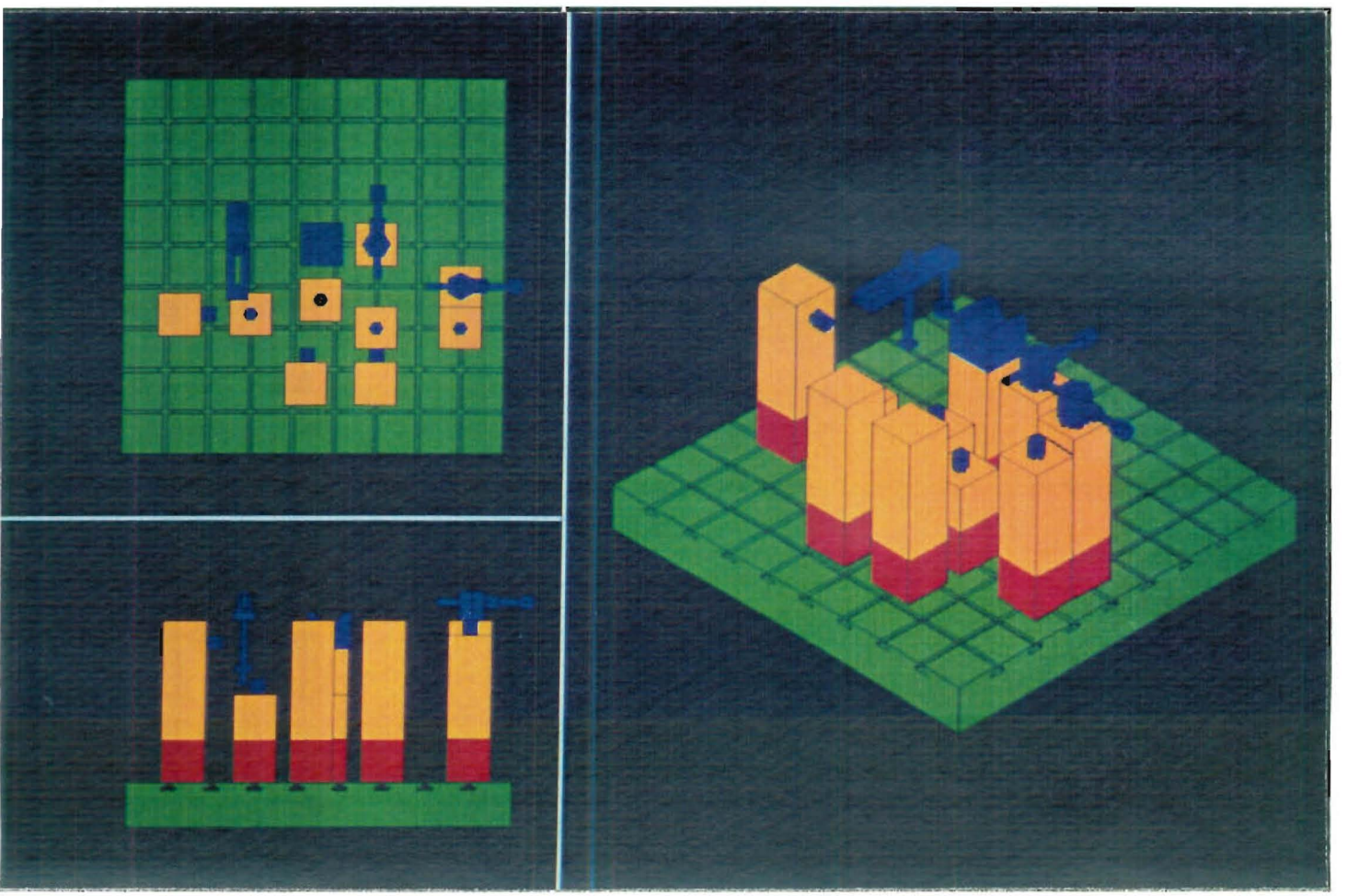


Figure 47 Vertical face towers added (workpiece removed)

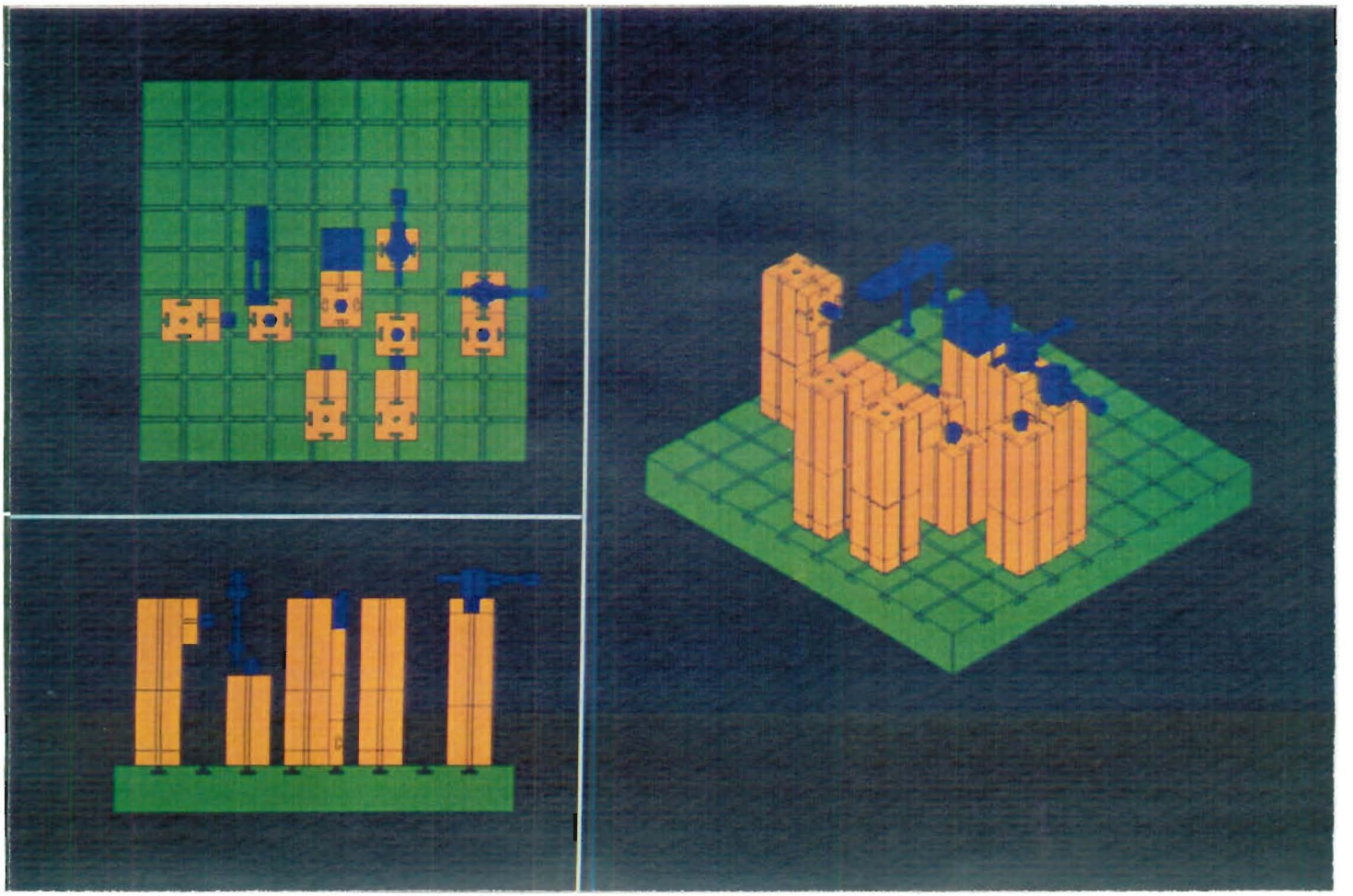


Figure 48 Final output (workpiece removed)

MATRIX SPATIAL REPRESENTATION OF THE OUTPUT

40	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
400	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
420	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
440	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
460	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
480	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
540	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

70	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
400	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
420	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
440	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
460	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
480	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
540	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

110	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
400	0	0	0	0	0	0	0	0	0	0	0	805	805	805	0	0	806	806	806	0	0	0	0	0	0	0	0
420	0	0	0	0	0	0	0	0	0	0	0	805	805	805	0	0	806	806	806	0	0	0	0	0	0	0	0
440	0	0	0	0	0	0	0	0	0	0	0	805	805	805	0	0	806	806	806	0	0	0	0	0	0	0	0
460	0	0	0	0	0	0	0	0	0	0	0	805	805	805	0	0	806	806	806	0	0	0	0	0	0	0	0
480	0	0	0	0	0	0	0	0	0	0	0	805	805	805	0	0	806	806	806	0	0	0	0	0	0	0	0
540	0	0	0	0	0	0	0	0	0	0	0	842	842	842	0	0	843	843	843	0	0	0	0	0	0	0	0

130	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
400	0	0	0	0	0	0	0	0	0	0	0	805	805	805	0	0	806	806	806	0	0	0	0	0	0	0	0
420	0	0	0	0	0	0	0	0	0	0	0	805	805	805	0	0	806	806	806	0	0	0	0	0	0	0	0
440	0	0	0	0	0	0	0	0	0	0	0	805	805	805	0	0	806	806	806	0	0	0	0	0	0	0	0
460	0	0	0	0	0	0	0	0	0	0	0	805	805	805	0	0	806	806	806	0	0	0	0	0	0	0	0
480	0	0	0	0	0	0	0	0	0	0	0	805	805	805	0	0	806	806	806	0	0	0	0	0	0	0	0
540	0	0	0	0	0	0	0	0	0	0	0	842	842	842	0	0	843	843	843	0	0	0	0	0	0	0	0

Figure 49(1)

150	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
400	0	0	0	0	0	0	0	0	0	0	0	901	901	901	0	0	902	902	902	0	0	0	0	0	0	0	0
420	0	0	0	0	0	0	0	0	0	0	0	901	901	901	0	0	902	902	902	0	0	0	0	0	0	0	0
440	0	0	0	0	0	0	0	0	0	0	0	901	901	901	0	0	902	902	902	0	0	0	0	0	0	0	0
460	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
480	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
540	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

170	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
400	0	0	0	0	0	0	0	0	0	0	0	901	901	901	0	0	902	902	902	0	0	0	0	0	0	0	0
420	0	0	0	0	0	0	0	0	0	0	0	901	901	901	0	0	902	902	902	0	0	0	0	0	0	0	0
440	0	0	0	0	0	0	0	0	0	0	0	901	901	901	0	0	902	902	902	0	0	0	0	0	0	0	0
460	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
480	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
540	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

190	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
400	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
420	0	0	0	0	0	0	0	0	0	0	0	0	626	0	0	0	0	626	0	0	0	0	0	0	0	0	0
440	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
460	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
480	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
540	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

210	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	632	632	632	632	632	632	632	632	632	632	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	0	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0
400	0	0	0	0	0	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	0	0	0	0	0
420	0	0	0	0	0	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	801	801	801	0	0
440	0	0	0	0	0	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	801	801	801	0	0
460	0	0	0	0	0	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	801	801	801	0	0
480	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	801	801	801	0	0
540	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	845	845	845	0	0	0	847	847	847	0	0

Figure 49(2)

230	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	632	632	632	632	632	632	632	632	632	632	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	0	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0
400	0	0	807	807	701	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	0	621	0	0	0
420	0	0	807	807	701	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	801	801	801	0	0
440	0	0	807	807	701	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	801	801	801	0	0
460	0	0	807	807	0	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	801	801	801	0	0
480	0	0	807	807	0	0	0	0	0	0	0	0	0	0	0	0	0	621	0	0	0	0	801	801	801	0	0
540	0	0	844	844	0	0	0	846	846	846	0	0	0	0	0	0	845	845	845	0	0	0	847	847	847	0	0

250	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	632	632	632	632	632	632	632	632	632	632	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	632	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0
400	0	0	807	807	701	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	0	0	0	0	0
420	0	0	807	807	701	628	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	801	801	801	0	0
440	0	0	807	807	701	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	801	801	801	0	0
460	0	0	807	807	0	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	801	801	801	0	0
480	0	0	807	807	0	0	0	0	0	621	0	0	0	0	0	0	0	0	0	0	0	0	801	801	801	0	0
540	0	0	844	844	0	0	0	0	846	846	846	0	0	0	0	0	845	845	845	0	0	0	847	847	847	0	0

270	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	632	632	632	632	632	632	632	632	632	632	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	632	0	632	632	632	632	632	632	632	632	632	632	0	0	0	0	0	0	0
400	0	0	807	807	701	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	802	802	802	0	0
420	0	0	807	807	701	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	802	802	802	0	0
440	0	0	807	807	701	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	802	802	802	0	0
460	0	0	807	807	0	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	802	802	802	0	0
480	0	0	807	807	0	0	0	0	621	0	0	0	0	621	0	0	0	0	0	0	0	0	802	802	802	0	0
540	0	0	844	844	0	0	0	846	846	846	0	0	941	941	941	0	0	0	0	0	0	0	848	848	848	0	0

290	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	632	632	632	632	632	632	632	632	632	632	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	632	0	632	632	632	632	632	632	632	632	632	632	632	632	632	621	632	0	0
400	0	0	0	0	0	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	802	802	802	0	0
420	0	0	0	0	0	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	802	802	802	0	0
440	0	0	0	0	0	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	802	802	802	0	0
460	0	0	0	0	0	0	632	632	632	632	632	632	632	632	632	632	632	632	632	632	0	0	802	802	802	0	0
480	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	802	802	802	0	0
540	0	0	0	0	0	0	0	0	0	0	0	0	941	941	941	0	0	0	0	0	0	0	848	848	848	0	0

Figure 49(3)

310	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	0	0	632	632	632	0	0	0	0	0	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	632	0	0	0	632	632	632	0	0	0	0	0	0	0	0	0	0	0	0
400	0	0	0	0	0	0	0	0	632	0	0	0	632	632	632	0	0	632	0	0	0	0	802	802	802	0	0
420	0	0	0	0	0	0	0	0	632	0	0	0	632	632	632	0	0	0	0	0	0	0	802	802	802	0	0
440	0	0	0	0	0	0	0	0	632	0	0	0	0	0	0	0	0	0	0	0	0	0	802	802	802	0	0
460	0	0	0	0	0	0	0	0	632	0	0	0	0	0	0	0	0	0	0	0	0	0	802	802	802	0	0
480	0	0	0	0	0	0	0	0	632	0	0	0	0	0	0	0	0	0	0	0	0	0	802	802	802	0	0
540	0	0	0	0	0	0	0	0	632	0	0	0	941	941	941	0	0	0	0	0	0	0	848	848	848	0	0

330	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	0	0	632	632	0	0	0	0	0	0	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	632	0	0	0	632	632	0	0	0	0	0	0	0	0	0	0	0	0	0
400	0	0	0	0	0	0	0	0	632	0	0	0	632	632	0	0	0	632	0	0	0	0	0	0	0	0	0
420	0	0	0	0	0	0	0	0	632	0	0	0	632	632	0	0	0	0	0	0	0	0	0	0	0	0	0
440	0	0	0	0	0	0	0	0	632	0	0	0	803	803	0	0	804	804	804	0	0	0	0	0	0	0	0
460	0	0	0	0	0	0	0	0	632	0	0	0	803	803	0	0	804	804	804	0	0	0	0	0	0	0	0
480	0	0	0	0	0	0	0	0	632	0	0	0	803	803	0	0	804	804	804	0	0	0	0	0	0	0	0
540	0	0	0	0	0	0	0	0	632	0	0	0	849	849	0	0	850	850	850	0	0	0	0	0	0	0	0

350	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	0	0	632	632	632	0	0	0	0	0	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	632	0	0	0	632	632	632	0	0	0	0	0	0	0	0	0	0	0	0
400	0	0	0	0	0	0	0	0	632	0	0	0	632	632	632	0	0	632	0	0	0	0	0	0	0	0	0
420	0	0	0	0	0	0	0	0	632	0	0	0	632	621	632	0	0	621	0	0	0	0	0	0	0	0	0
440	0	0	0	0	0	0	0	0	632	0	0	0	803	803	803	0	804	804	804	0	0	0	0	0	0	0	0
460	0	0	0	0	0	0	0	0	632	0	0	0	803	803	803	0	804	804	804	0	0	0	0	0	0	0	0
480	0	0	0	0	0	0	0	0	632	0	0	0	803	803	803	0	804	804	804	0	0	0	0	0	0	0	0
540	0	0	0	0	0	0	0	0	632	0	0	0	849	849	849	0	850	850	850	0	0	0	0	0	0	0	0

370	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	0	0	632	632	632	0	0	0	0	0	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	632	0	0	0	632	632	632	0	0	0	0	0	0	0	0	0	0	0	0
400	0	0	0	0	0	0	0	0	632	0	0	0	632	632	632	0	0	632	0	0	0	0	0	0	0	0	0
420	0	0	0	0	0	0	0	0	632	0	0	0	632	632	632	0	0	0	0	0	0	0	0	0	0	0	0
440	0	0	0	0	0	0	0	0	632	0	0	0	803	803	803	0	804	804	804	0	0	0	0	0	0	0	0
460	0	0	0	0	0	0	0	0	632	0	0	0	803	803	803	0	804	804	804	0	0	0	0	0	0	0	0
480	0	0	0	0	0	0	0	0	632	0	0	0	803	803	803	0	804	804	804	0	0	0	0	0	0	0	0
540	0	0	0	0	0	0	0	0	632	0	0	0	849	849	849	0	850	850	850	0	0	0	0	0	0	0	0

Figure 49(4)

900	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
400	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
420	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
440	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
460	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
480	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
540	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

940	20	70	90	130	150	170	190	210	230	250	270	290	310	330	350	370	390	410	430	450	470	490	510	530	550	900	920
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
380	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
400	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
420	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
440	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
460	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
480	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
540	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

To show the whole matrix clearly, only the last 3 digits of the spatial numbers are displayed.

solid space	632
horizontal face locator (32621)	621
thrust locator - north (32626)	626
thrust locator - east (32628)	628
modular blocks - x direction(32701-32799)	701-799
modular blocks - y direction(32801-32899)	801-899
modular blocks - x direction(32901-32999)	901-999

Figure 49(5)

```
# include "ciminimum.h"
# include "cimacros.h"
char r[5];
double
pt1[3],pt2[3],pt3[3],pt4[3],pt5[3],
pt6[3],pt7[3],pt8[3],pt9[3],pt10[3],
pt11[3],pt12[3],pt13[3],pt14[3],pt15[3],
pt16[3],pt17[3],pt18[3],pt19[3],pt20[3],
pt21[3],pt22[3],pt23[3],pt24[3],pt25[3],
pt26[3],pt27[3],pt28[3],pt29[3],pt30[3],
pt31[3],pt32[3],pt33[3],pt34[3],pt35[3],
pt36[3],pt37[3],pt38[3],pt39[3],pt40[3],
pt41[3],pt42[3],pt43[3],pt44[3],pt45[3],
pt46[3],pt47[3],pt48[3],pt49[3],pt50[3],
pt51[3],pt52[3],pt53[3],pt54[3],pt55[3],
pt56[3],pt57[3],pt58[3],pt59[3],pt60[3],
pt61[3],pt62[3],pt63[3],pt64[3],pt65[3],
pt66[3],pt67[3],pt68[3],pt69[3],pt70[3],
pt71[3],pt72[3],pt73[3],pt74[3],pt75[3],
pt76[3],pt77[3],pt78[3],pt79[3],pt80[3],
pt81[3],pt82[3],pt83[3],pt84[3],pt85[3],
pt86[3],pt87[3],pt88[3],pt89[3],pt90[3],
pt91[3];
main()
{
ci$put(cmd = "co");
ci$put(string = "green");
ci$put(cmd = "wt");
ci$put(value = 4);
ci$put(cmd = "st");
ci$put(value = 3);
ci$put(cmd = "ly");
ci$put(value = 2);
ci$put(cmd = "define active cell library");
ci$put(string = "realblk");
ci$put(cmd = "define active cell");
ci$put(string = "y120");
pt1[0] = 490;
pt1[1] = 190;
pt1[2] = -400;
ci$put(cmd = "place cell");
ci$put(point = pt1);
ci$put(cmd = "define active cell");
ci$put(string = "y20");
pt2[0] = 490;
pt2[1] = 190;
pt2[2] = -520;
ci$put(cmd = "place cell");
ci$put(point = pt2);
```

Figure 50(1)

```
ci$put(cmd = "define active cell");
ci$put(string = "y120");
pt3[0] = 490;
pt3[1] = 250;
pt3[2] = -380;
ci$put(cmd = "place cell");
ci$put(point = pt3);
ci$put(cmd = "define active cell");
ci$put(string = "y40");
pt4[0] = 490;
pt4[1] = 250;
pt4[2] = -500;
ci$put(cmd = "place cell");
ci$put(point = pt4);
ci$put(cmd = "define active cell");
ci$put(string = "y120");
pt5[0] = 290;
pt5[1] = 310;
pt5[2] = -420;
ci$put(cmd = "place cell");
ci$put(point = pt5);
ci$put(cmd = "define active cell");
ci$put(string = "y120");
pt6[0] = 370;
pt6[1] = 310;
pt6[2] = -420;
ci$put(cmd = "place cell");
ci$put(point = pt6);
ci$put(cmd = "define active cell");
ci$put(string = "y120");
pt7[0] = 270;
pt7[1] = 70;
pt7[2] = -380;
ci$put(cmd = "place cell");
ci$put(point = pt7);
ci$put(cmd = "define active cell");
ci$put(string = "y40");
pt8[0] = 270;
pt8[1] = 70;
pt8[2] = -500;
ci$put(cmd = "place cell");
ci$put(point = pt8);
ci$put(cmd = "define active cell");
ci$put(string = "y120");
pt9[0] = 370;
pt9[1] = 70;
pt9[2] = -380;
ci$put(cmd = "place cell");
ci$put(point = pt9);
```

Figure 50(2)

```
ci$put(cmd = "define active cell");
ci$put(string = "y40");
pt10[0] = 370;
pt10[1] = 70;
pt10[2] = -500;
ci$put(cmd = "place cell");
ci$put(point = pt10);
ci$put(cmd = "define active cell");
ci$put(string = "y120");
pt11[0] = 70;
pt11[1] = 210;
pt11[2] = -380;
ci$put(cmd = "place cell");
ci$put(point = pt11);
ci$put(cmd = "define active cell");
ci$put(string = "y40");
pt12[0] = 70;
pt12[1] = 210;
pt12[2] = -500;
ci$put(cmd = "place cell");
ci$put(point = pt12);
ci$put(cmd = "define active cell");
ci$put(string = "y60");
pt13[0] = 370;
pt13[1] = 190;
pt13[2] = -480;
ci$put(cmd = "place cell");
ci$put(point = pt13);
ci$put(cmd = "define active cell");
ci$put(string = "y60");
pt14[0] = 190;
pt14[1] = 210;
pt14[2] = -480;
ci$put(cmd = "place cell");
ci$put(point = pt14);
ci$put(cmd = "define active cell");
ci$put(string = "x20");
pt15[0] = 130;
pt15[1] = 210;
pt15[2] = -380;
ci$put(cmd = "place cell");
ci$put(point = pt15);
ci$put(cmd = "define active cell");
ci$put(string = "z40");
pt16[0] = 270;
pt16[1] = 130;
pt16[2] = -380;
ci$put(cmd = "place cell");
ci$put(point = pt16);
```

Figure 50(3)

```
ci$put(cmd = "define active cell");
ci$put(string = "z40");
pt17[0] = 370;
pt17[1] = 130;
pt17[2] = -380;
ci$put(cmd = "place cell");
ci$put(point = pt17);
ci$put(cmd = "define active cell");
ci$put(string = "z60");
pt18[0] = 290;
pt18[1] = 250;
pt18[2] = -480;
ci$put(cmd = "place cell");
ci$put(point = pt18);
ci$put(cmd = "define active cell");
ci$put(string = "base540");
pt19[0] = 40;
pt19[1] = 40;
pt19[2] = -540;
ci$put(cmd = "place cell");
ci$put(point = pt19);
ci$put(respond = TERMINATE);
}
```

Figure 50(4)

```
**************************************
          Partlist
**************************************

  TYPE : MODULAR BLOCK 1
  Orientation of the block = y
  Length of the block = 120
  Placement coordinate :
     x = 490
     y = 190
     z = -400
  TYPE : MODULAR BLOCK 2
  Orientation of the block = y
  Length of the block = 20
  Placement coordinate :
     x = 490
     y = 190
     z = -520
  TYPE : MODULAR BLOCK 3
  Orientation of the block = y
  Length of the block = 120
  Placement coordinate :
     x = 490
     y = 250
     z = -380
  TYPE : MODULAR BLOCK 4
  Orientation of the block = y
  Length of the block = 40
  Placement coordinate :
     x = 490
     y = 250
     z = -500
  TYPE : MODULAR BLOCK 5
  Orientation of the block = y
  Length of the block = 120
  Placement coordinate :
     x = 290
     y = 310
     z = -420
  TYPE : MODULAR BLOCK 6
  Orientation of the block = y
  Length of the block = 120
  Placement coordinate :
     x = 370
     y = 310
     z = -420
  TYPE : MODULAR BLOCK 7
  Orientation of the block = y
  Length of the block = 120
  Placement coordinate :
     x = 270
     y = 70
     z = -380
```

Figure 51(1)

```
TYPE : MODULAR BLOCK 8
Orientation of the block = y
Length of the block = 40
Placement coordinate :
   x = 270
   y = 70
   z = -500
TYPE : MODULAR BLOCK 9
Orientation of the block = y
Length of the block = 120
Placement coordinate :
   x = 370
   y = 70
   z = -380
TYPE : MODULAR BLOCK 10
Orientation of the block = y
Length of the block = 40
Placement coordinate :
   x = 370
   y = 70
   z = -500
TYPE : MODULAR BLOCK 11
Orientation of the block = y
Length of the block = 120
Placement coordinate :
   x = 70
   y = 210
   z = -380
TYPE : MODULAR BLOCK 12
Orientation of the block = y
Length of the block = 40
Placement coordinate :
   x = 70
   y = 210
   z = -500
TYPE : MODULAR BLOCK 13
Orientation of the block = y
Length of the block = 60
Placement coordinate :
   x = 370
   y = 190
   z = -480
TYPE : MODULAR BLOCK 14
Orientation of the block = y
Length of the block = 60
Placement coordinate :
   x = 190
   y = 210
   z = -480
```

Figure 51 (2)

```
TYPE : MODULAR BLOCK 15
Orientation of the block = x
Length of the block = 20
Placement coordinate :
   x = 130
   y = 210
   z = -380
TYPE : MODULAR BLOCK 16
Orientation of the block = z
Length of the block = 40
Placement coordinate :
   x = 270
   y = 130
   z = -380
TYPE : MODULAR BLOCK 17
Orientation of the block = z
Length of the block = 40
Placement coordinate :
   x = 370
   y = 130
   z = -380
TYPE : MODULAR BLOCK 18
Orientation of the block = z
Length of the block = 60
Placement coordinate :
   x = 290
   y = 250
   z = -480
TYPE : BASEPLATE
Size of the baseplate = 540mm X 540mm
Placement coordinate :
   x = 40
   y = 40
   z = -540
```

Figure 51(3)

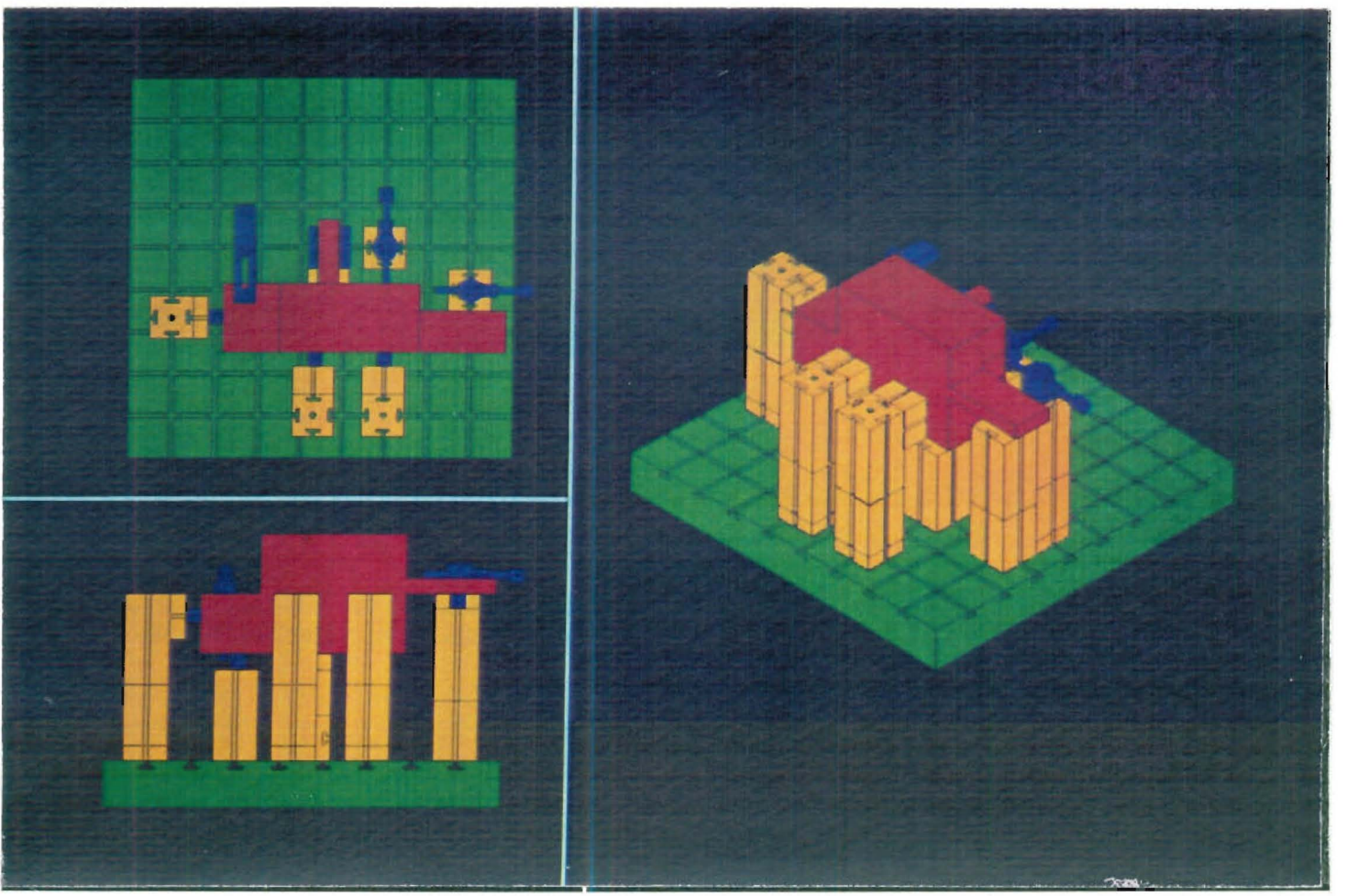


Figure 52 Final solution

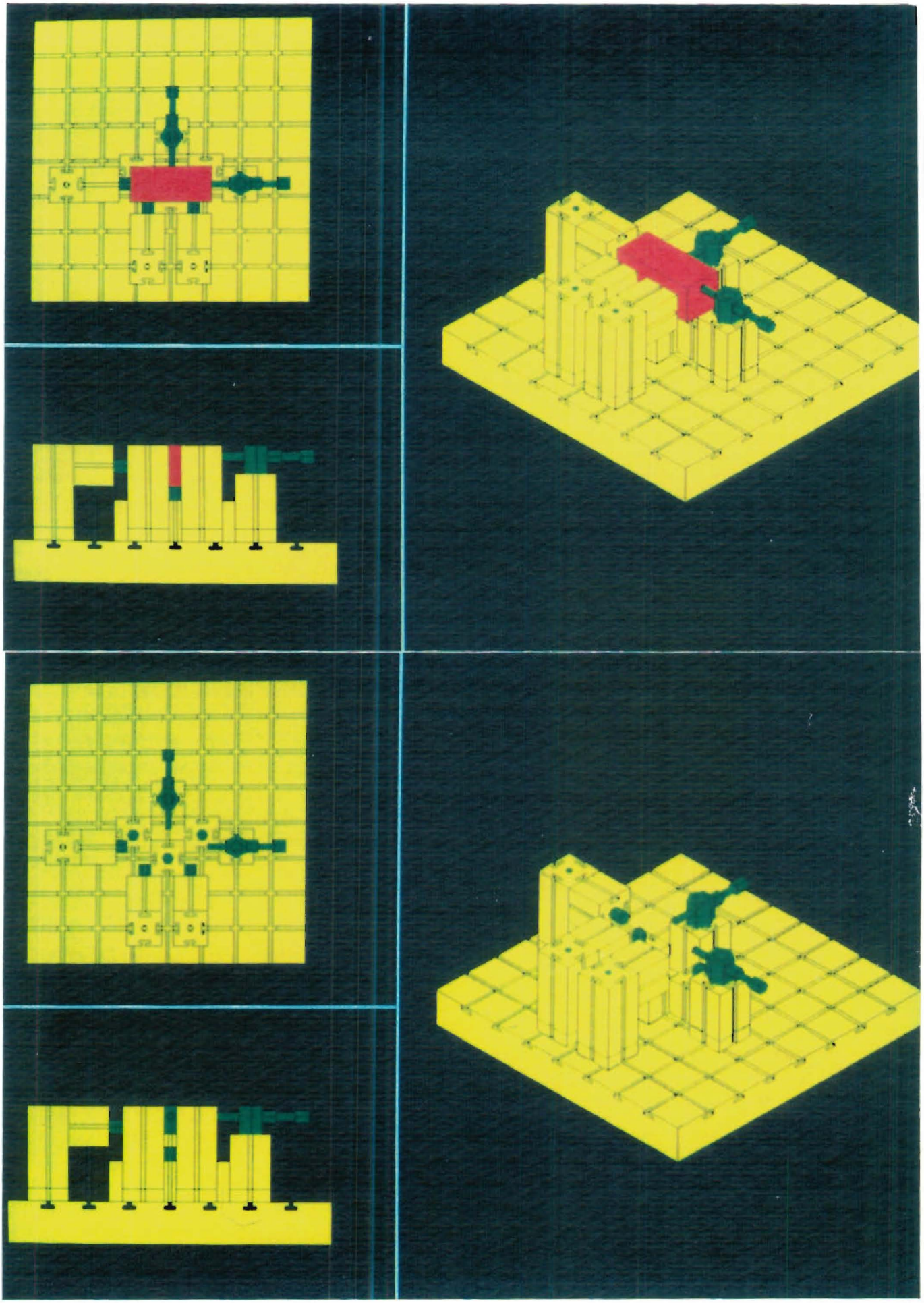


Figure 53 Work example

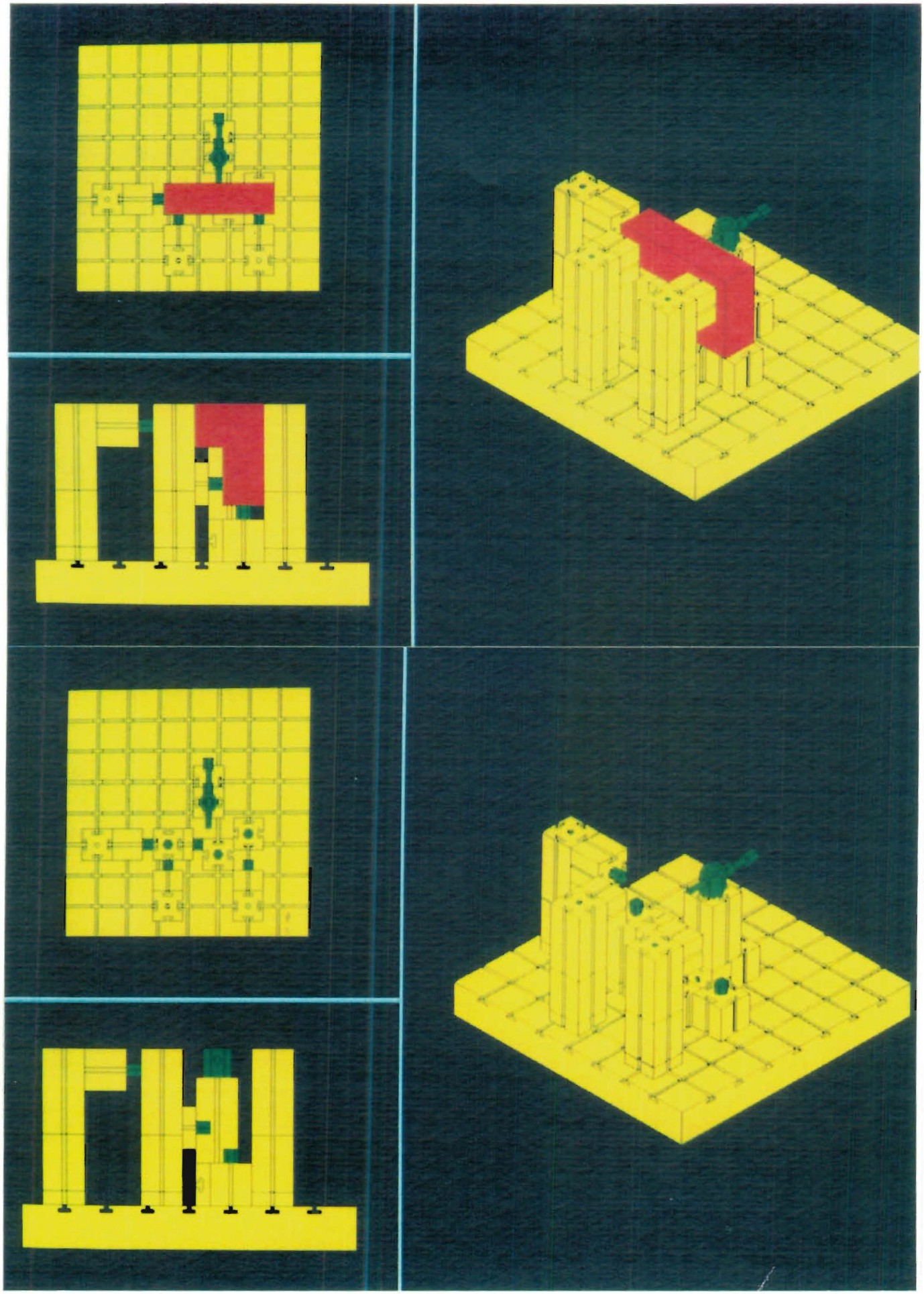


Figure 54 Work example

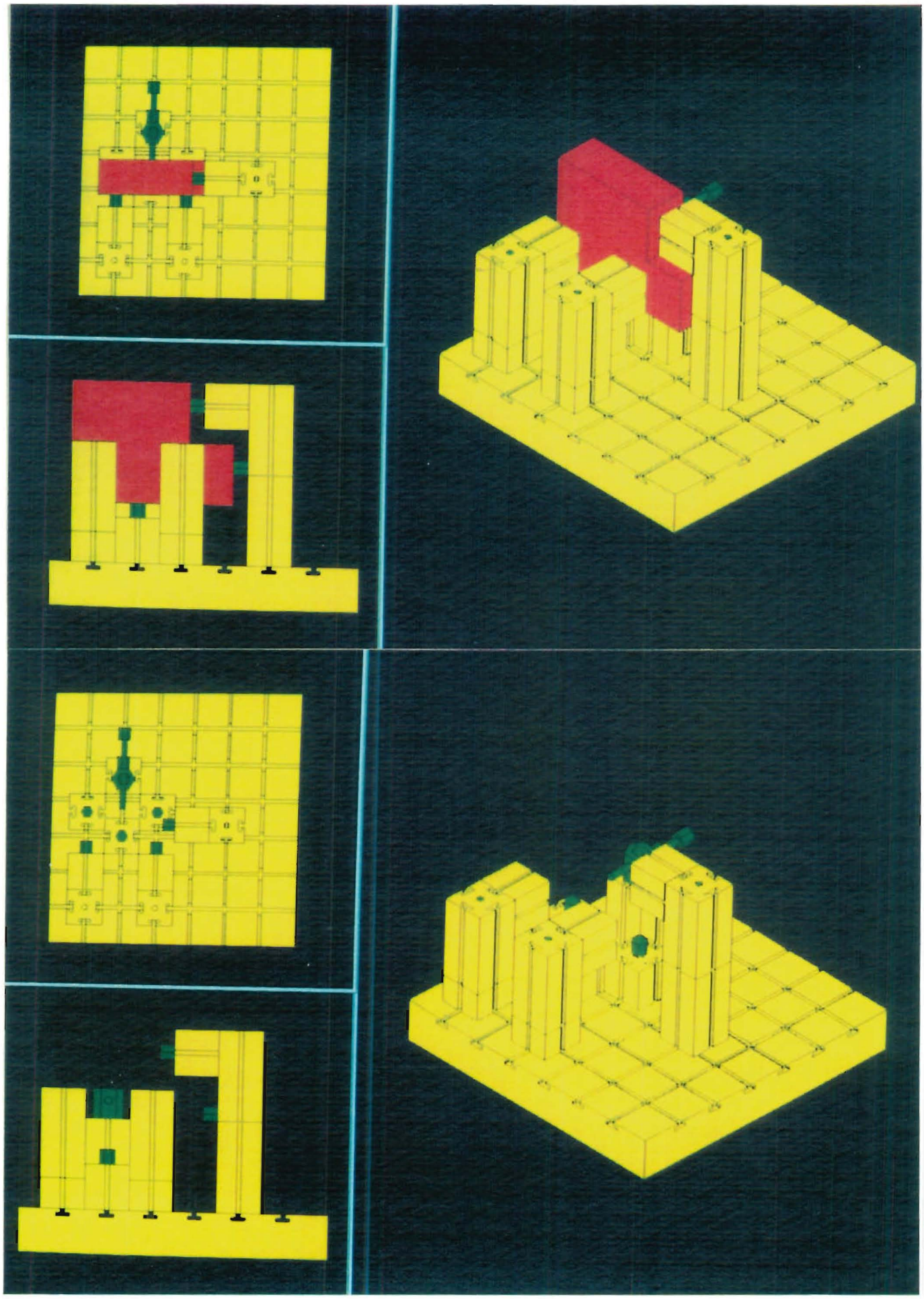


Figure 55 Work example

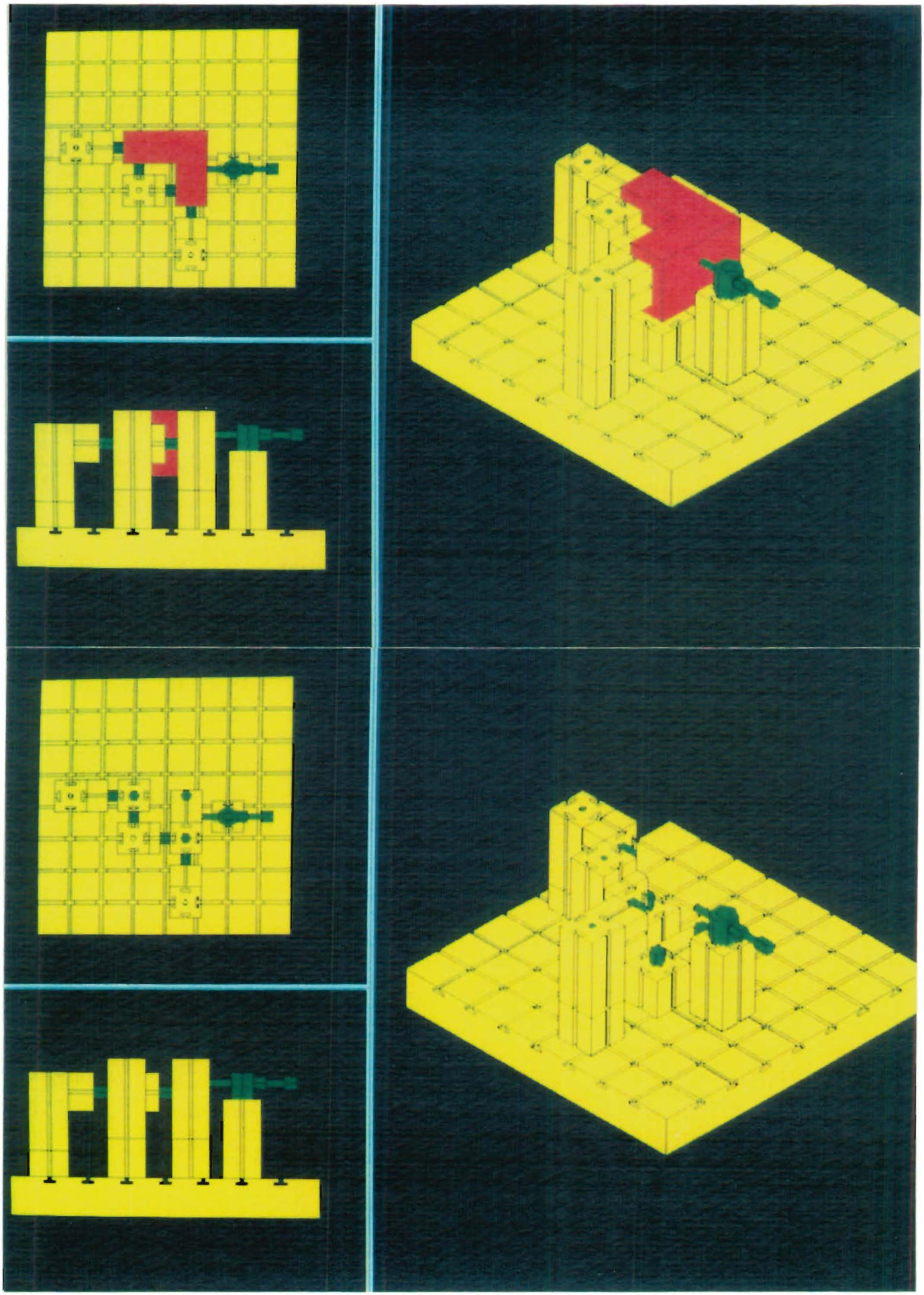


Figure 56 Work example

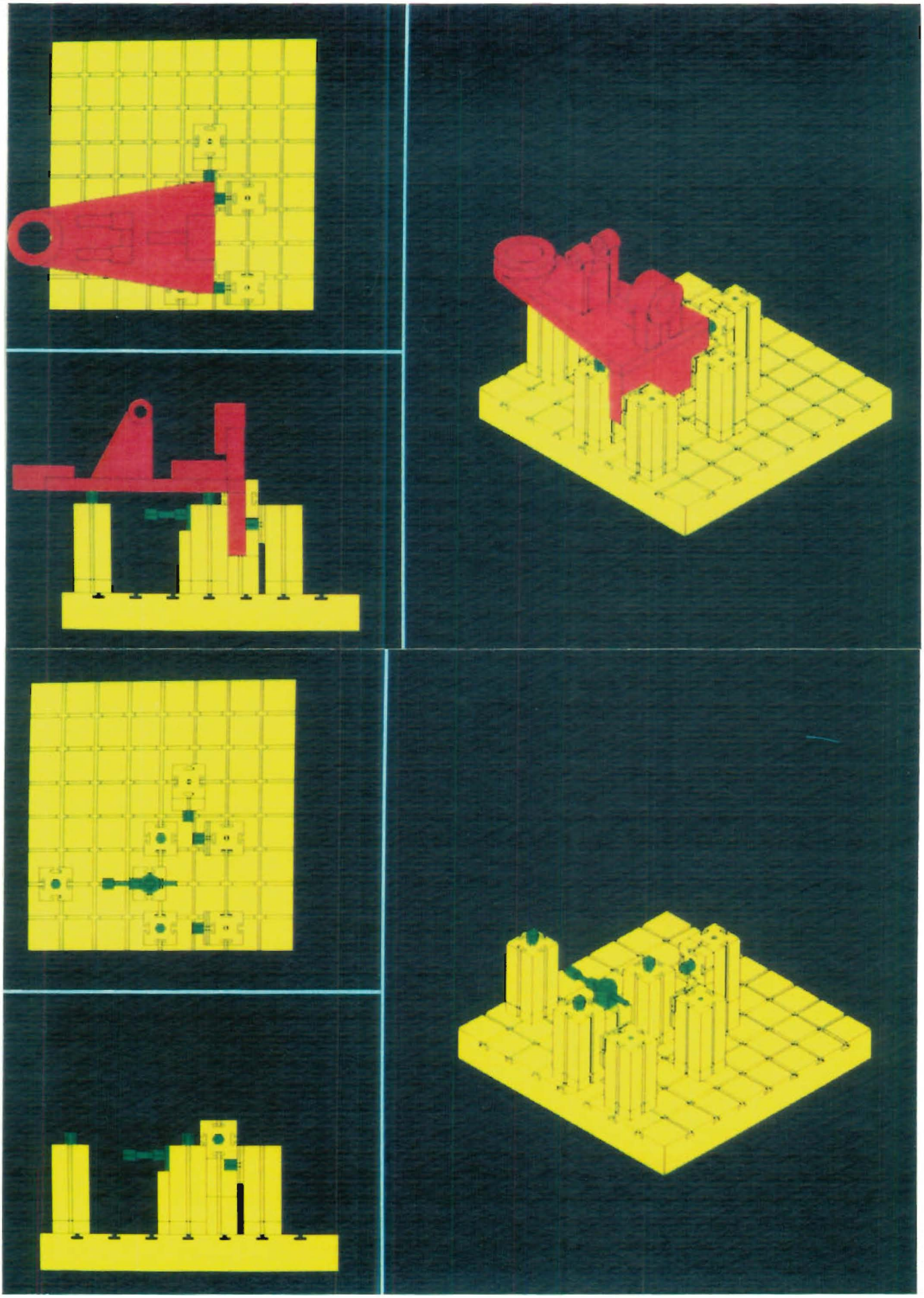


Figure 57 Work example

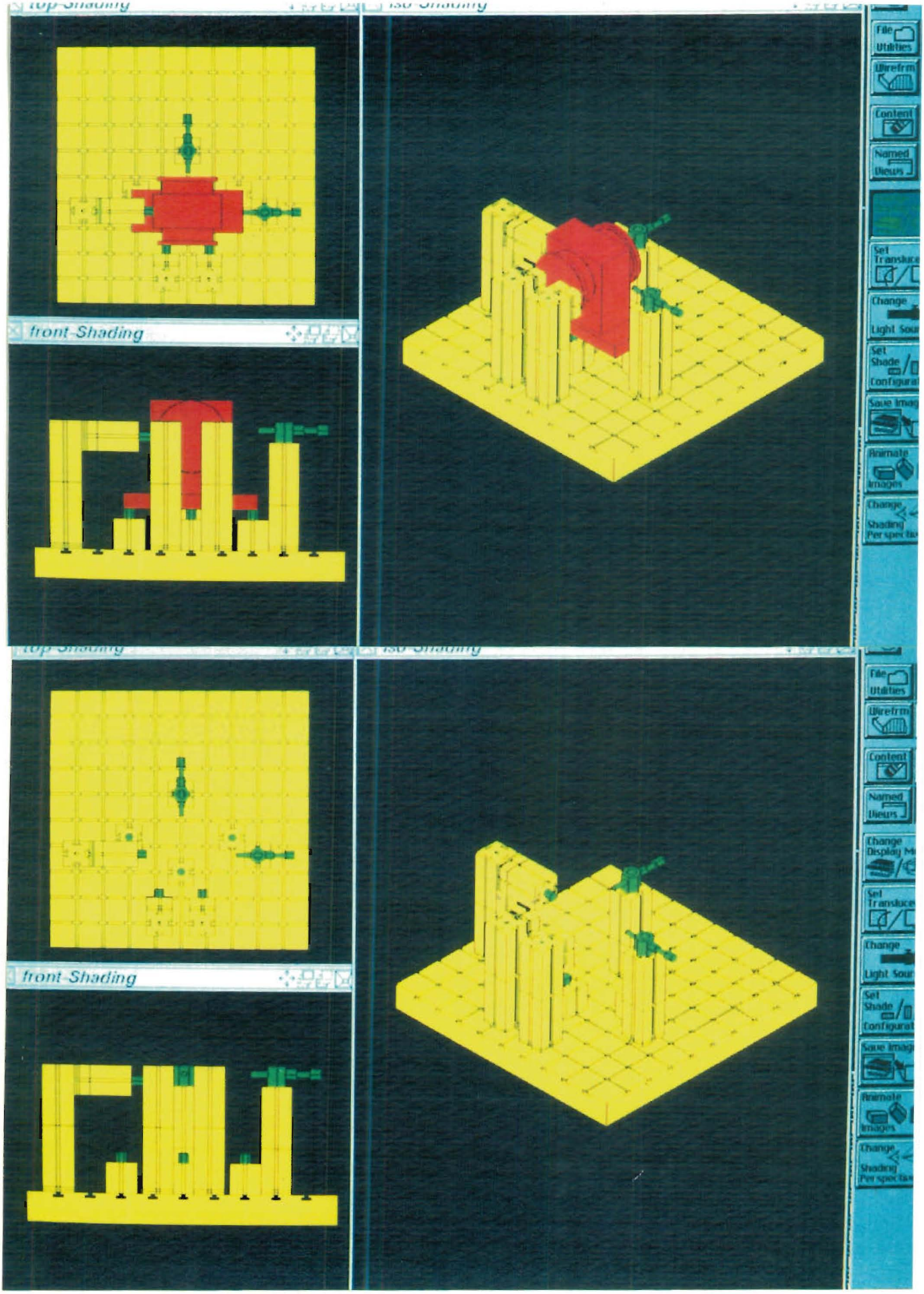


Figure 58 Work example

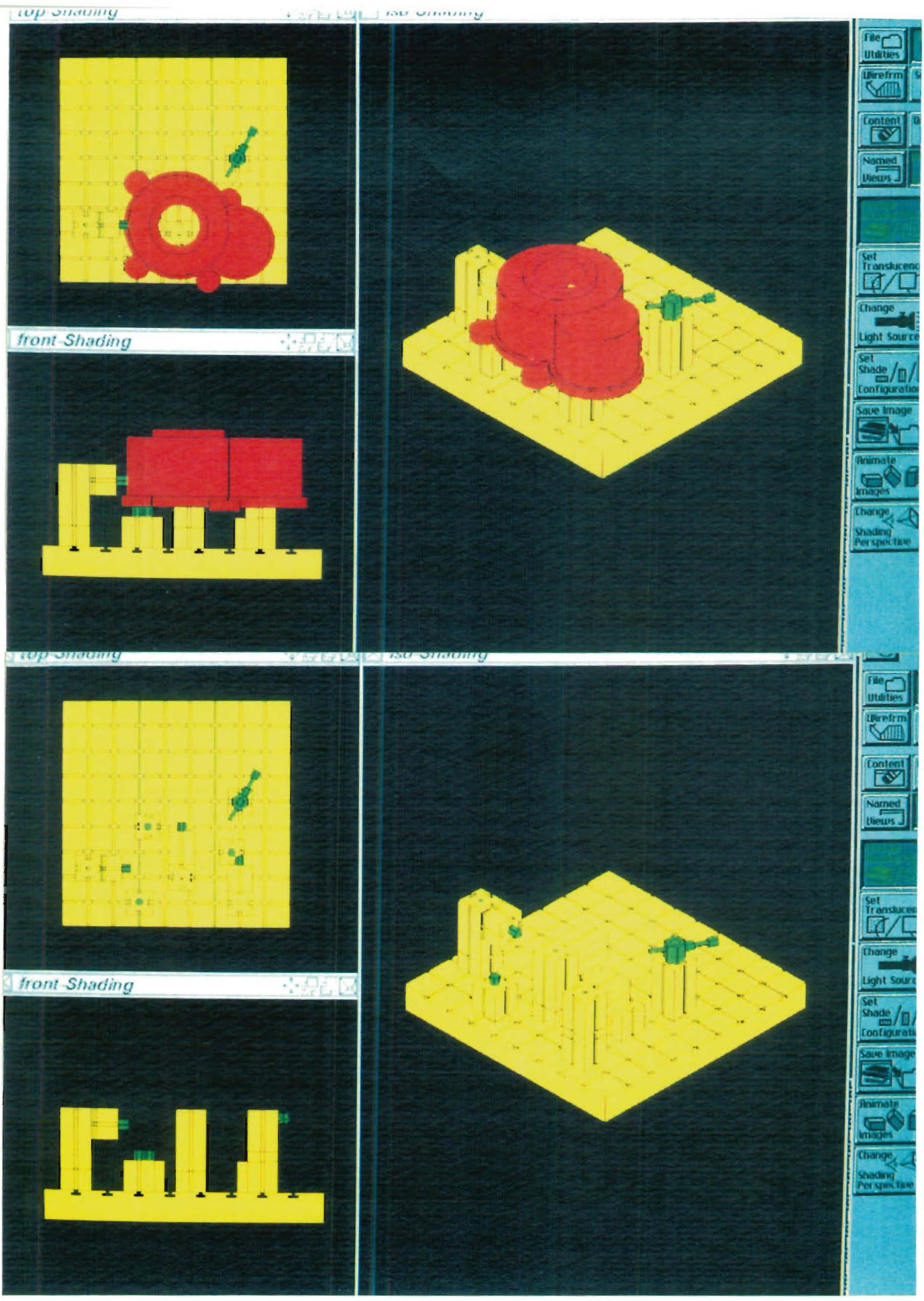


Figure 59 Work example

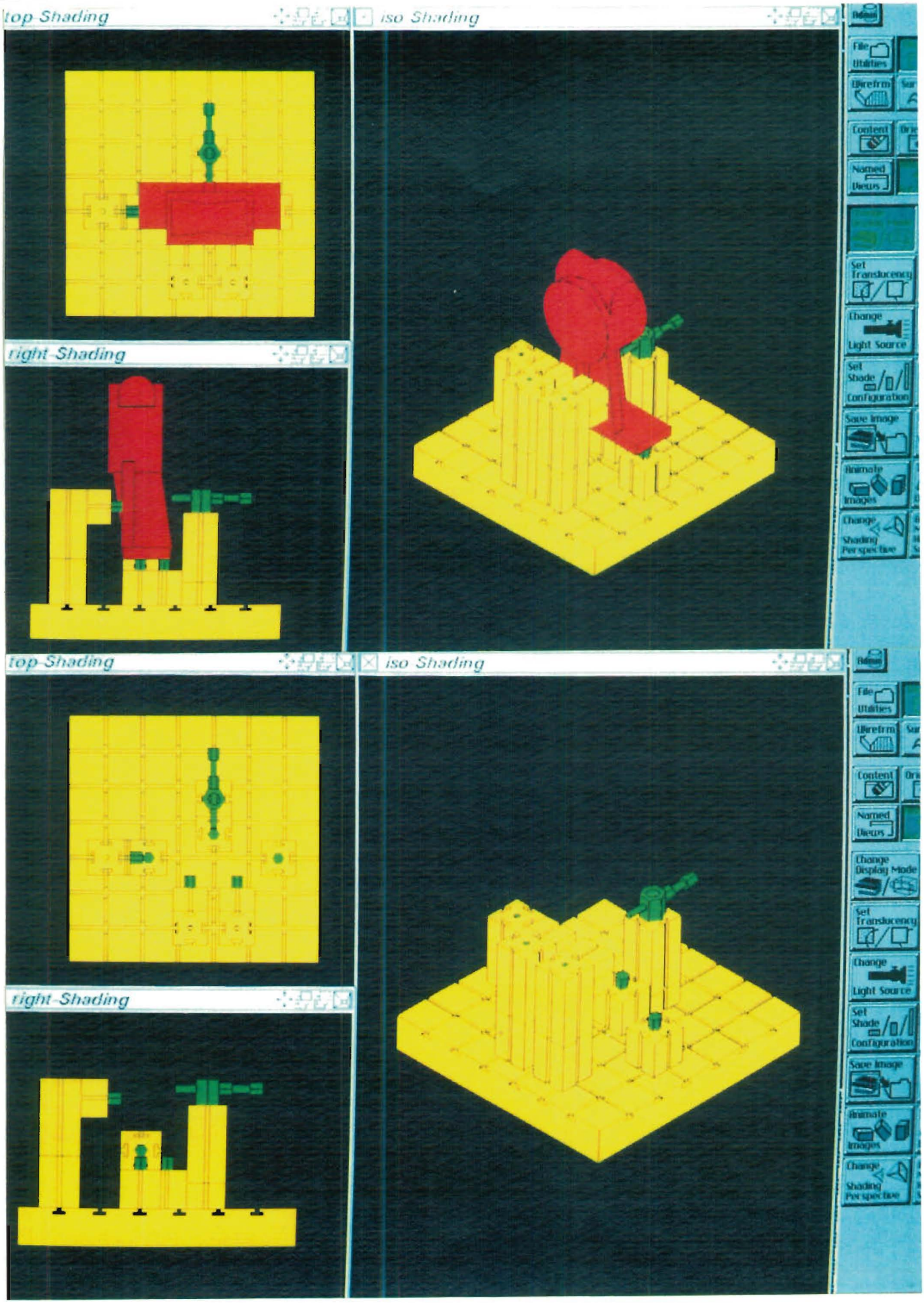


Figure 60 Work example

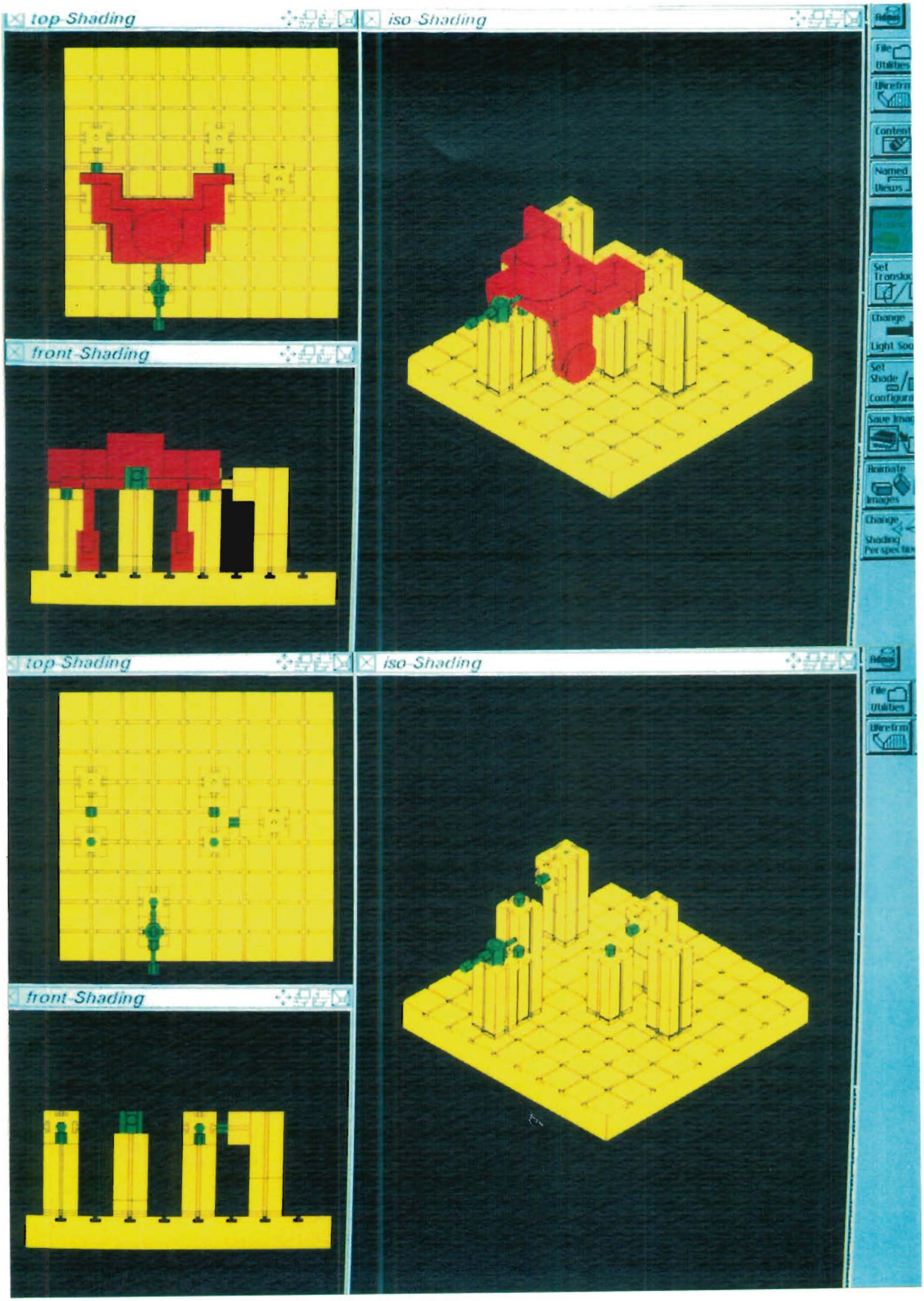


Figure 61 Work example

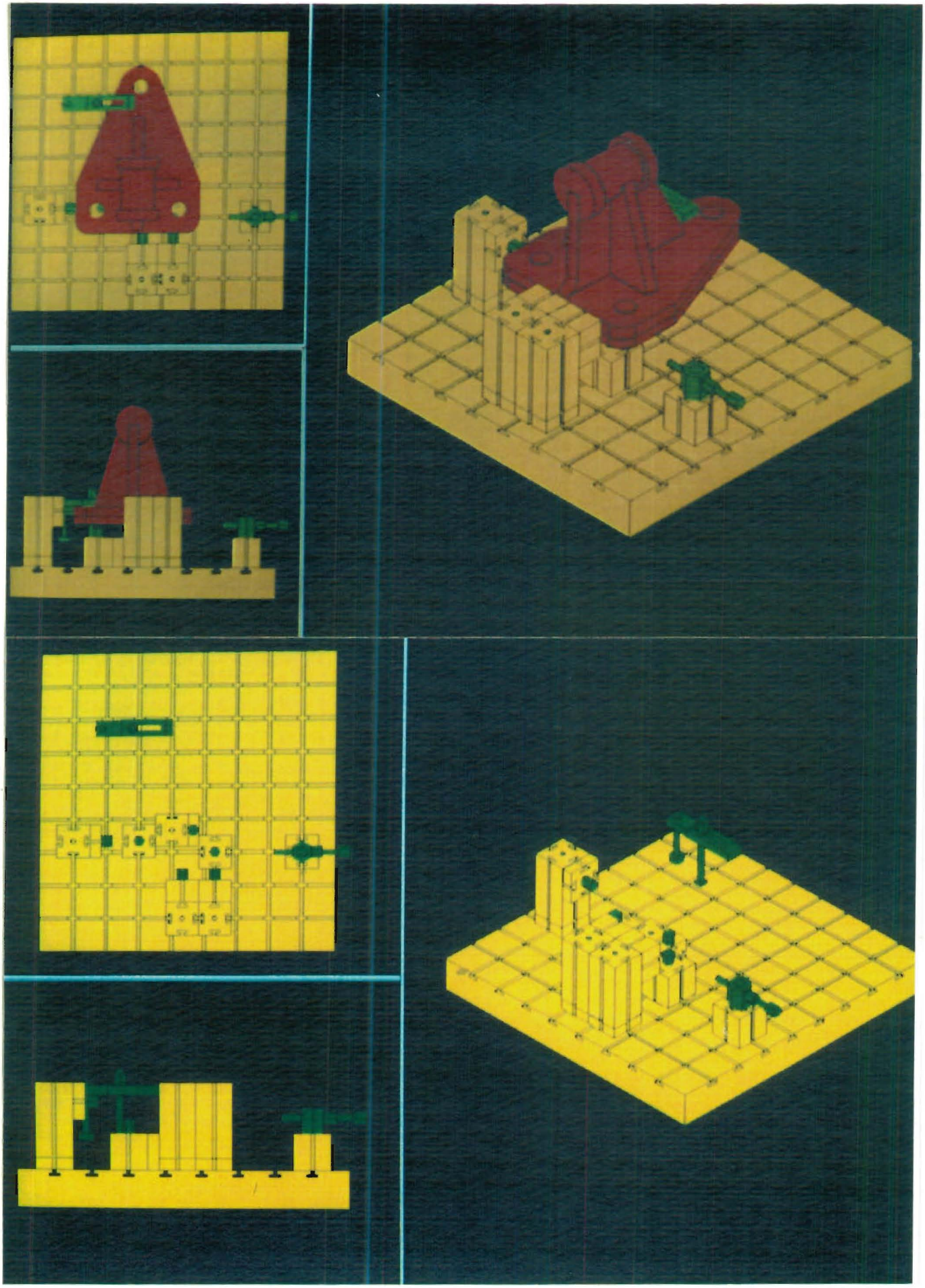


Figure 62 Work example

CHAPTER 11

CONCLUSION & FUTURE WORK

The programs demonstrate that the objective of this research has been achieved. The procedures developed here are implemented on an IBM AT computer, and hence can readily be made available to designers. The output is a simple form, a list of standard components and their position in the design space. This information can be used as input to a simple parametric program to enable display and drawing production on any CAD system. Manual design methods are difficult to document. CAD provides full documentation of the fixture design enabling accurate rebuilds to be made, and this in turn will encourage the recycling of fixture components.

This work introduces an efficient algorithm for the assembly of modular fixture elements. This method designs the fixture body using a systematic procedure. A key feature of the method is the development of a suitable spatial representation technique, which has also demonstrated its capability to represent and manipulate the fixture components effectively. Unlike the other workers, this method is deterministic; therefore this work provides an alternative method for automating the modular fixture assembly process. It is also a valuable step towards automatic design of fixtures with great possibilities of practical applications.

The automated design of the assembly of modular fixtures shortens the time to reconfigure the fixture, and therefore brings FMS one step closer to achieving long term flexibility. Further work is in progress to develop an expert system to provide a link between process planning and fixture design, which will aid in the choice of location, clamping and support elements.

BIBLIOGRAPHY

[1] M.P.Groover, *Automation, Production Systems, and Computer Aided Manufacturing*, Prentice-Hall, New Jersey, 1980.

[2] D.J.Williams, *Manufacturing Systems*, Halsted Press, New York-Toronto, and Open University Press, Milton Keynes, 1987.

[3] M.P.Groover and E.W.Jr.Zimmers, *CAD/CAM : Computer Aided Design and Manufacturing*, Prentice-Hall, New Delhi, 1986.

[4] J.E.Lenz, *Flexible Manufacturing*, Marcel Dekker, New York, 1989.

[5] J.Talavage and R.G.Hannam, *Flexible Manufacturing Systems in Practice*, Marcel Dekker, New York, 1988.

[6] J.Mortimer (ed.), *The FMS Report*, IFS, Bedford, England.

[7] D.F.Eary and G.E.Johnson, *Process Engineering for Manufacturing*, Prentice-Hall, New Jersey, 1962.

[8] E.K.Hendriksen, *Jigs and Fixture Design Manual*, Industrial Press, New York, 1973.

[9] P.C.Sharma, *Production Engineering*, S Chand & Co., Ram Nagar, New Delhi, 1982.

[10] H.Sedlik, *Jigs & Fixtures*, Society of Manufacturing Engineers, Dearborn, Michigan, 1970.

[11] F.D.Jones, *Jig and Fixture Design*, Allied Publishers, India.

[12] E.J.H.Jones and H.C.Town, *Production Engineering - Jig and Fixture Design*, Butterworth & Co., London, 1972.

[13] F.W.Wilson (ed.), *Handbook of Fixture Design*, McGraw-Hill, New York, 1962.

[14] M.H.A.Kempster, *An Introduction to Jig and Tool Design*, 3rd Edition, Hodder and Stoughton, London, 1974.

[15] M.H.A.Kempster, *Principle of Jig and Tool Design*, Hart Publishing Company, New York.

[16] C.Donaldson, G.H.LeCain and V.C.Goold, *Tool Design*, Tata McGraw-Hill, New Delhi, 1973.

[17] W.E.Boyes (ed.), *Jigs and Fixtures*, Society of Manufacturing Engineers, Michigan, 1982.

[18] G.H.Ryder, *Jigs, Fixtures, Tools and Gauges*, Technical Press, London, 1973.

[19] K.Whybrew, R.J.Astley and B.K.A.Ngoi, "Computer Aided Assembly of Modular Fixturing Systems", *International Conference on Industrial Engineering*, Singapore, September 1988.

[20] B.S.Lim and J.A.G.Knight, "HOLDEX - Holding device expert system", *Proceedings of the First International Conference on Applications of Artificial Intelligence in Engineering Problems*, Southampton, UK, volume 1, April 1986, pp 483-501.

[21] B.K.A.Ngoi, K.Whybrew and R.J.Astley, "Computer Aided Assembly of Modular Fixturing Systems", *IA'90 Conference*, Singapore, May 1990.

[22] M.V.Gandhi and B.S.Thompson, "The Integration of CAD and CAM in Adaptive Fixturing for Flexible Manufacturing Systems", *Conference Proceedings, ASME International Computers in Engineering*, August 1985, pp 301-305.

[23] P.M.Ferreira, B.Kochar, C.R.Liu and V.Chandru, "AIFIX: An Expert System Approach to Fixture Design", *Winter Annual Meeting of the American Society of Mechanical Engineers*, Miami Beach, USA, November 1985.

[24] A.Markus, "Strategies for the Automated Generation of Modular Fixtures", *Proceedings of Manufacturing International*, Atlanta, USA, volume 3, April 1988, pp 97-103.

[25] A.Markus, Z.Markusz, J.Farkus and J.Filemon, "Fixture Design Using Prolog: An Expert System", *Robotic and Computer Integrated Manufacturing*, Volume 1, No.2, 1984, pp 167-172.

[26] M.V.Gandhi and B.S.Thompson, "Automated Design of Modular Fixtures for Flexible Manufacturing Systems", *Journal of Manufacturing Systems*, Volume 5, No.4, 1986, pp 243-252.

[27] A.S.Miller and R.G.Hannam, "Computer aided design using a knowledge base approach and its application to the design of jigs and fixtures", *Proc Instn Mech Engrs*, Volume 199, No.B4, 1985, pp 227-234.

[28] A.Y.C.Nee, N.Bhattacharyya and A.N.Poo, "A knowledge-based CAD of jigs and fixtures", Tech. Paper No.TE85-902, *Society of Manufacturing Engineers*, USA, 1985.

[29] P.M.Grippo, M.V.Gandhi and B.S.Thompson, "The Computer Aided Design of Modular Fixturing Systems", *The International Journal of Advanced Manufacturing Technology*, Volume 2, No.2, 1987, pp 75-88.

[30] D.T.Pham, M.J.Nategh and A.de Sam Lazaro, "A knowledge based Jig and Fixture Designers' Assistant", *The International Journal of Advanced Manufacturing Technology*, Volume 4, No.1, 1989, pp 26-45.

[31] A.Y.C.Nee, N.Bhattacharyya and A.N.Poo, "Applying AI in Jig and Fixture Design", *Robotics and Computer-Integrated Manufacturing*, Volume 3, No.2, 1987, pp 195-200.

[32] Y.C.Chou and M.M.Barash, "Computerized Fixture Design from Solid Models of Workpieces", *Winter Annual Meeting of the American Society of Mechanical Engineers*, Anaheim, CA, USA, Volume 21, Dec 1986, pp 133-141.

[33] A.de Sam Lazaro, D.T.Pham and F.J.Robinson, "Automated Design of Fixtures for FMS", *International Conference on Computer Integrated Manufacturing*, USA, 1988, pp 201-207.

[34] J.R.Woodward and D.Graham, "Automated assembly and inspection of versatile fixtures", *Second International Conference on Flexible Manufacturing Systems. IFS (Conferences)*, London, October 1983, pp 425-30.

[35] B.S.Thompson, "Flexible Fixturing - A Current Frontier in the Evolution of Flexible Manufacturing Cells", *The American Society of Mechanical Engineers*, ASME Paper No. 84-WA/Prod-16, 1984.

[36] B.S.Thompson and M.V.Gandhi, "A Commentary on Flexible Fixturing", *Applied Mechanics Reviews*, Volume 39, No.10, October 1986, pp 1365-1369.

[37] M.V.Gandhi and B.S.Thompson, "Phase-change fixturing for FMS", *Manufacturing Engineering*, Volume 93, No.6, December 1984, pp 79-80.

[38] P.K.Wright, E.Kurokawa and M.R.Cutkosky, "Programmable conformable clamps", *AUTOFACT 4 Conference Proceedings*, SME, Dearbon, Michigan, 1982, pp 11.51-11.58.

[39] M.V.Gandhi and B.S.Thompson, "Phase-change fixturing for flexible manufacturing systems", *Journal of Manufacturing Systems*, Volume 4, No..1, 1985, pp 29-39.

[40] H.Asada and A.B.By, "Kinematic Analysis of Workpiece Fixturing for Flexible Assembly with Automatically Reconfigurable Fixtures", *IEEE Journal of Robotics and Automation*, RA-1(2), pp 86-94, June 1985.

[41] J.E.Biegel, "The Future of Artificial Intelligence (AI) in Manufacturing", *Computers & Industrial Engineering*, Volume II, No.1-4, 1986, pp 276-279.

[42] D.Ben-Arieh, "Industrial Engineering Application on Expert System", *Computers & Industrial Engineering*, Volume II, No.1-4, 1986, pp 459-463.

[43] W.Myers, "Introduction of Expert Systems", *IEEE Expert*, Spring 1986, pp 100-109.

[44] P.J.Denning, "Towards a Science of Expert Systems", *IEEE Expert*, Summer 1986, pp 80-83.

[45] C.L.Dym, "Expert Systems : New Approaches to Computer-aided Engineering", *Engineering with Computers*, Spring 1985, pp 9-25.

[46] W.Eversheim and R.Neitzel, "Expert Systems for Flexible Manufacturing", *Second International Summer Seminar*, Dubrovnik, Yugoslavia, August 1987, pp 67-85.

[47] L.Alting and H.C.Zhang, "Computer Aided Process Planning : the state-of-the-art survey", *International Journal of Production Research*, Volume 27, No.4, 1989, pp 553-585

[48] R.G.Hannam and A.S.Miller, "Linking Design and Manufacture through CADCAM - a case study on jig fixture design", *Effective CADCAM 1985*, Cambridge, England, July 1985, pp 117-124.

[49] B.S.Lim and J.A.G.Knight, "A Foundation for a Knowledge-base Computer Integrated Manufacturing System", *Artificial Intelligence*, Volume 2, No.1, 1987, pp 11-22.

[50] J.E.Biegel, "The Future Role of Expert Systems in Manufacturing", *Computers & Industrial Engineering*, Volume II, No.1-4, 1986, pp 473-475.

[51] P.J.Englert and P.K.Wright, "Application of Artificial Intelligences and the design of fixtures for automated manufacturing", *IEEE International Conference on Robotics and Automation*, San Francisco, CA, USA, Volume 1, April 1986, pp 345-351.

[52] G.Lewis, "Modular Fixturing System", *Proceedings of the 2nd International Conference on Flexible Manufacturing Systems*, London, October 1983, pp 451-462.

[53] E.G.Hoffman, *Modular Fixturing*, Manufacturing Technology Press, Lake Geneva, Wisconsin, 1987.

[54] E.Nebergall, "What About Modular Fixturing?", *Modern Machine Shop*, Volume 60, No.6, November 1987, pp 54-64.

[55] T.Horie, "Adaptability of a Modular Fixturing System to Factory Automation", *Bull Japan Society of Prec Engg*, Volume 22, No.1, March 1988, pp 1-5.

[56] *Catic Modular Fixture System*, catalogue from George Kuikka Ltd, Watford, UK.

[57] *Halder Modular Jig and Fixture System*, catalogue from Halder, West Germany.

[58] M.Montalbano, *Decision Tables*, Science Research Associates, 1974.

[59] H.McDaniel, *Applications of Decision Tables*, Brandon Systems Press, 1970.

[60] E.Humby, *Programs from Decision Tables*, MacDonald, London and New York, 1973.

[61] A.P.Perovskii, "Design Testing and Reliability of Machines Universal Grippers for Industrial Robots", *Russian Engineering Journal*, Volume 60, No.8, 1980, pp 9-11.

[62] D.Scott, "Amazing Hardening Fluid", *Popular Science*, April 1984, pp 82-85.

[63] *Turbo Pascal*, Owner's Handbook, Version 4.0. Borland International, Scotts Valley, USA.

[64] *Programmable Clamping*, a brochure by The Mors group, 23, avenue Aristide briand, B.P.108 PARON - 89104 SENS - FRANCE.

[65] "Internal Representation of 3D Geometric Models", lecture notes by A. Parkinson, Cambridge University Engineering Department,1984.

[66] R.E.Crowley, "Solid Models - What are They?", Tech. Paper No.MS83-742, *Society of Manufacturing Engineers*, USA, 1983.

[67] A.A.G.Requicha and H.B.Voelcker,"Solid Modelling; A Historical Summary and Contemporary Assessment", *IEEE Computer Graphics and Applications*, Volume 2, No.2, 1982, pp 9-24.

[68] A.A.G.Requicha and H.B.Voelcker,"Solid Modelling: Current Status and Research Direction", *IEEE Computer Graphics and Applications*, Volume 3, No.7, 1983, pp 25-37.

[69] M.J.Pratt,"Solid Modelling and the Interface between Design and Manufacture", *IEEE Computer Graphics and Applications*, Volume 4, July 1984, pp 52-59.

[70] K.Yamaguchi,T.L.Kunii and K.Fujimura,"Octree-Related Data Structures and Algorithms", *IEEE Computer Graphics and Applications*, Volume 4, No.1, 1984, pp 53-59.

[71] K.Whybrew, G.A.Britton, D.F.Robinson and Y.Sermsuti-Anuwat, "A Graph-theoretic Approach to Tolerance Charting", *The International Journal of Advanced Manufacturing Technology*, Volume 5, No.2, 1990, pp 175-183.

[72] B.Wordenweber,"Automatic Mesh Generation of 2 and 3 Dimensional Curvilinear Manifolds", Ph.D Thesis, Computer Laboratory, University of Cambridge, 1981.

[73] TIPS Working Group, "TIPS-1 Technical Information Processing System", Institute of Precision Engineering, Hokkaido University, Sapporo 060, Japan.

[74] K Yamaguchi et al., "Computer-Integrated Manufacturing of Surfaces Using Octree Encoding", *IEEE Computer Graphics and Applications*, Vol 4, No 1, Jan 1984, pp 60-65.

[75] G C Carey and A de Pennington, "A Study of the Interface Between CAD and CAM Using Geometric Modelling Techniques", *Proc Anglo-Hungarian Sem Computational Geometry for CAD/CAM*, Cambridge, Sept 1983, G E M Jared and T Varady, eds., Cambridge University Engineering Department.

[76] Meagher D, "Geometric Modelling Using Octree Encoding; *Computer Graphic and Image Processing*, Vol 19, 1982, pp 129-147.

APPENDIX A

FLEXIBLE FIXTURING

Flexible fixturing involves employing a single fixturing system which holds parts of various shapes and sizes when subjected to the wide variety of external force fields and torques associated with manufacturing operations. Flexible fixtures may be classified into 3 categories:

(a) adaptable fixtures,
(b) reconfigurable fixtures, and
(c) "sculptured-surface" fixtures.

FMS usually employs many adaptable fixtures to handle workpieces from the various part families it is designed to handle. These fixtures contribute to the short term flexibility of FMS. (The concept of flexibility and reconfigurability of FMS has been discussed in section 1.1.3).

However, when a workpiece to be processed does not belong to any of the part families which the adaptable fixtures are designed for, it cannot be held by the existing adaptable fixtures. In such situation a reconfigurable fixture is required. A reconfigurable fixture should ideally be able to reconfigure itself instantly to fixture workpieces of any shapes and sizes. In practice a truly reconfigurable fixture does not exist. Many solutions, which have different degrees of

reconfigurability, have been proposed. Some of these solutions are outlined in section 2 (reconfigurable fixtures).

A different class of fixture is used to hold workpieces with sculptured surfaces. Such fixtures are usually very versatile in adapting to the surfaces of workpieces. However there is generally a constraint on the size and shape of the workpiece the fixture can accommodate.

FLEXIBLE FIXTURING : SOME SOLUTIONS

The following sections study a number of proposed solutions for flexible fixturing. Some solutions can only be classified as workholding devices because these solutions provide support without location. Therefore such a device requires a separate location system to define the workpiece position and orientation. It is important to note that some solutions are not tested.

1 Adaptable fixtures/workholding device

Adaptable fixtures/workholding devices usually require little or no design and manufacturing lead time. These fixtures/workholding devices are generally designed to hold parts belonging to a pre-defined family. This section outlines some solutions in this category.

Multi-leaf vise [35]

Petal collet and the multi-leaf vise are two very simple workholding devices described by B S Thompson in his review paper on flexible fixturing [35]. Multi-leaf vise consists of a fixed jaw and a moveable jaw as shown in Figure 63. The moveable jaw is of constant geometry while the fixed jaw is made up of multi leaves. These leaves are pivoted on a rod and can only rotate about the rod. The rotational movement is restrained by torsional springs, thus allowing workpiece of different geometries to be held.

Advantages

1 The multi-leaf vise can hold parts belonging to the part family it is designed to hold without any fixture design and manufacture lead time.

2 It is a simple and inexpensive device.

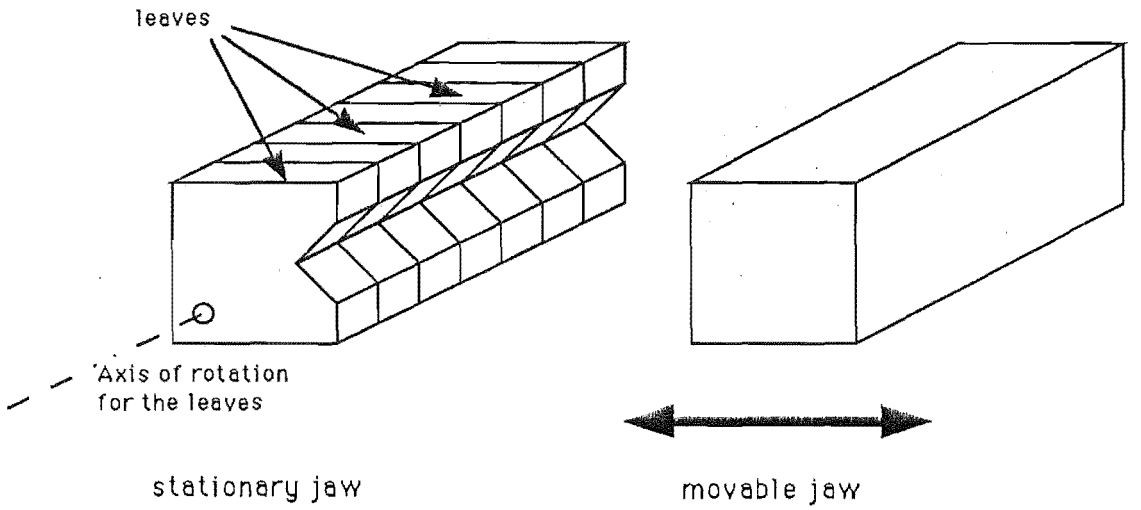
Disadvantages

1 The multi leaves can only hold part of certain shape and size.

2 The clamping action of the leaves may disturb the orientation of the workpiece to be clamped.

3 Metal chips may jam between the leaves preventing the vise from functioning properly.

4 The device needs a separate location system to define the workpiece position and orientation.



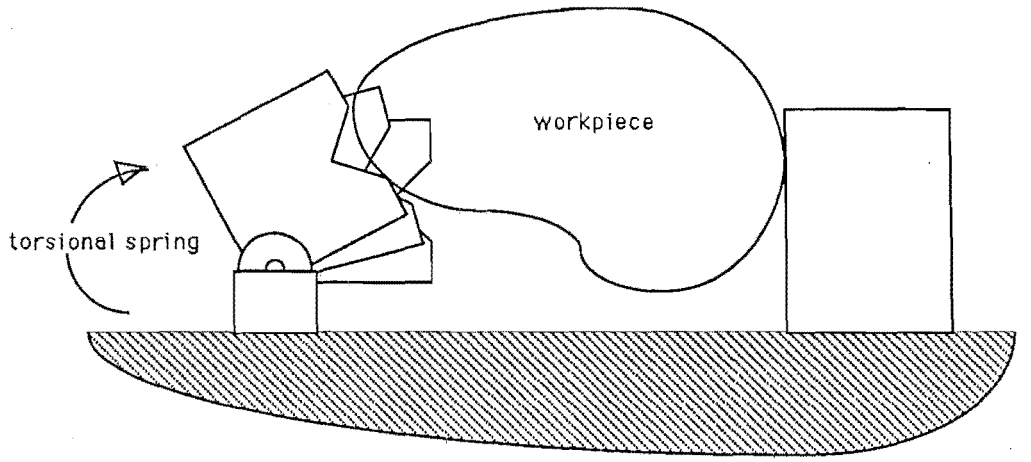


Figure 63 Multi-leaf vise

Petal collet [35]

Petal collet clamping device is made up of petals which are connected at a common joint as shown in Figure 64. The workpiece is positioned on the axis of symmetry of the device. The clamping ring then moves upwards to hold the workpiece in position. This device is simple but may not adequately hold the workpiece in position.

Advantages

1 The petal collet device, like the multi-leaf vise, does not require any fixture design and manufacture lead time, as long as the part introduced belongs to the part family it is designed to hold.

2 It is a simple and inexpensive device.

Disadvantages

1 The petal collet device can only hold part of certain shape and size.

2 The clamping action of the device is not positive, therefore it may result in part not adequately constrained or even damages to the surface of the part.

3 The movement of clamping ring upward to grasp the workpiece may disturb the orientation of the workpiece.

4 Metal chips may jam between the petals preventing the collet from functioning properly.

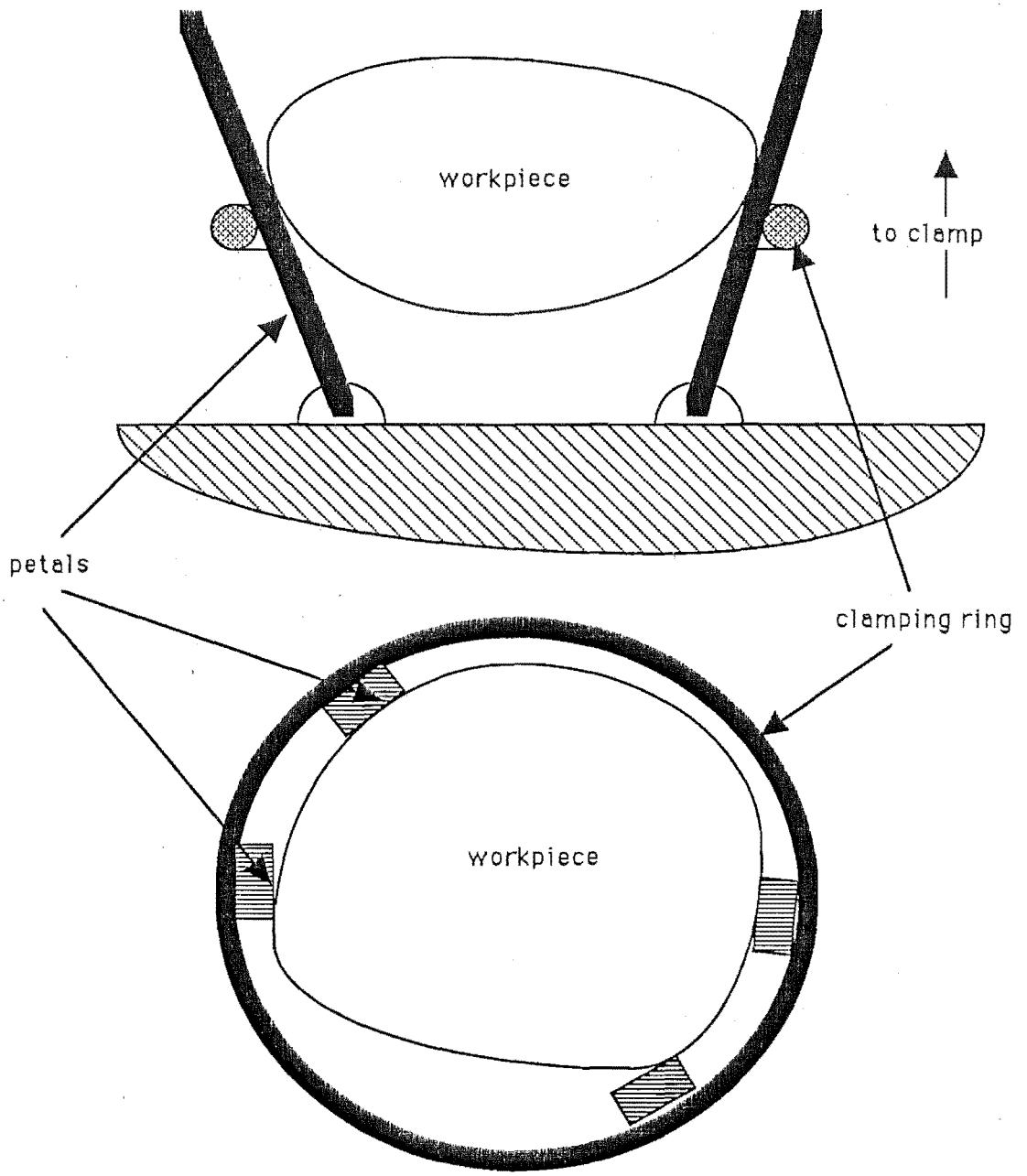


Figure 64 Petal collet

5 The device needs a separate location system to define the workpiece position and orientation.

Programmable clamping [64]

This is a self-contained pallet system developed by the Mors group (figure 65). The main feature of the system is the programmable hydraulic clamping system. A microprocessor is used to control the various clamping cycles and adjust the clamping pressure. Each pallet may be programed individually using an interactive programming language. The data are then conveyed to the pallet by infrared beam.

Advantages

1 The programmable clamping system allows automatic loading and unloading of the workpiece.

2 It also allows automatic and precise adjustment of the clamping pressure.

Disadvantages

1 It is a relatively expensive system.

2 The device needs a separate fixturing system to locate and support the workpiece.

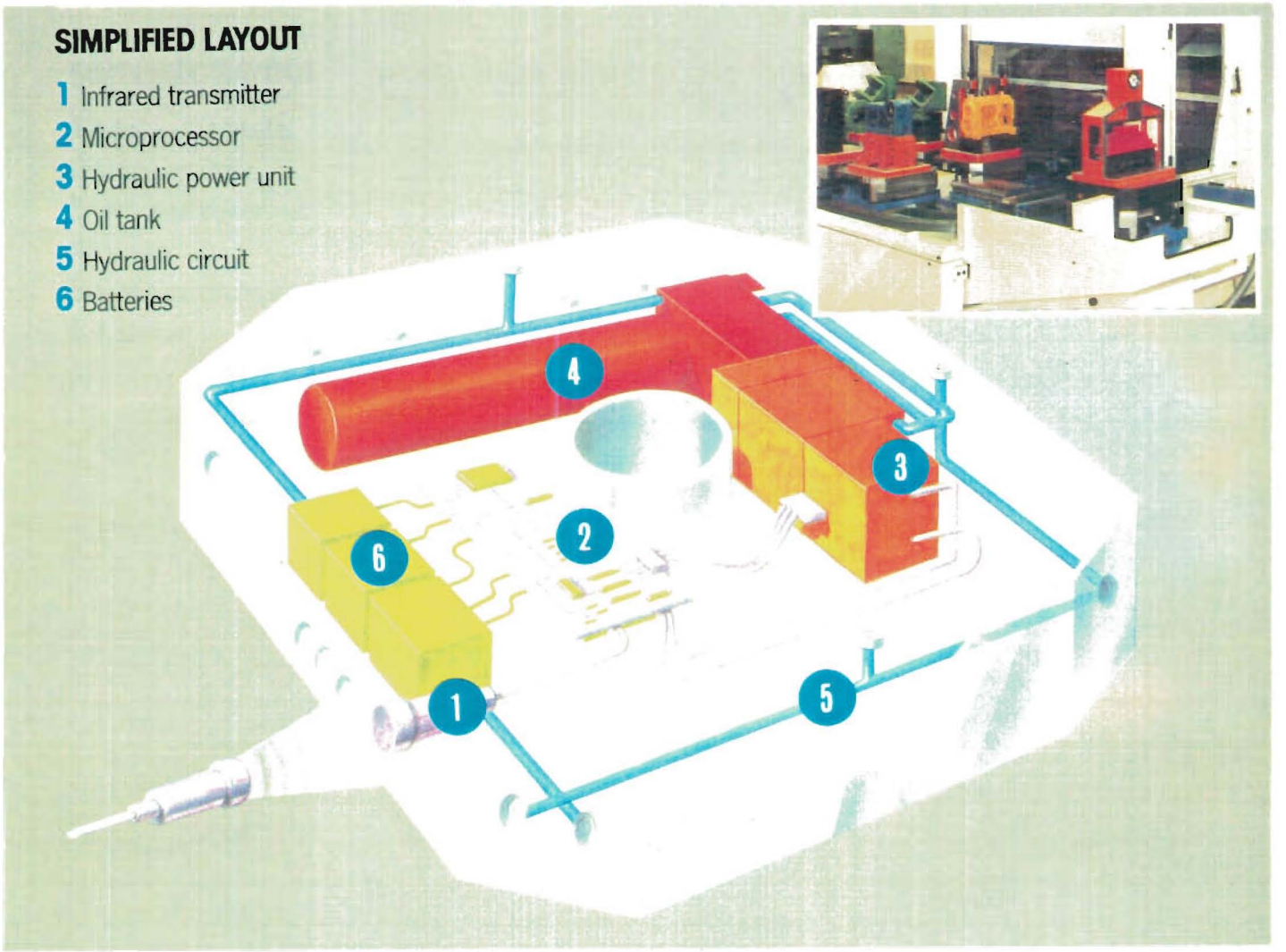


Figure 65 Programmable clamping system
(from a brochure by Mors Group [47])

2 Reconfigurable fixtures

To achieve true flexibility (both long term and short term flexibility), the flexible manufacturing system needs a truly reconfigurable fixture. This section reviews a number of solutions which have achieved different degrees of reconfigurability.

Modular fixturing system [52,53,56,57]

Recently, work on modular fixturing systems has formed a significant part of investigations into reconfigurable fixturing systems. The modular fixturing system has been described in chapter 1. It is built up from a combination of fixturing elements . A large variety of configurations can be obtained by using different combinations of these elements. Chapter 3 has discussed the design and manufacture concepts of modular fixturing system.

Advantages

1 Modular fixturing system is versatile and reliable.

2 Construction of the fixtures, given the assembly drawing, can be done in a few hours [52].

3 Modular fixturing elements can also be re-used and depending on their mechanical properties, can last for 15 to 20 years, which can reduce the capital investment by as much as 80% [26].

Disadvantages

1 The modular fixturing system lacks systematic assembly design procedure, therefore the assembly process can be quite complicated and disorganised without the assembly drawing.

2 The modular fixturing system is not suitable for robotic assembly.

Special-washer fixture [34]

The building block concept (Figure 66) taken by the University of Bath is specially designed for simple handling and assembly, to make the robotic assembly possible.

The fixture is constructed from vertical stacks of washers held on to the base plate by columns. The end of each stack is a locating or clamping element.

Advantages

1 The assembly process is simple and robotic assembly is possible.

2 The simple design, without complicated mechanisms, makes the fixture reliable.

Disadvantages

1 It is not as versatile as conventional modular fixture.

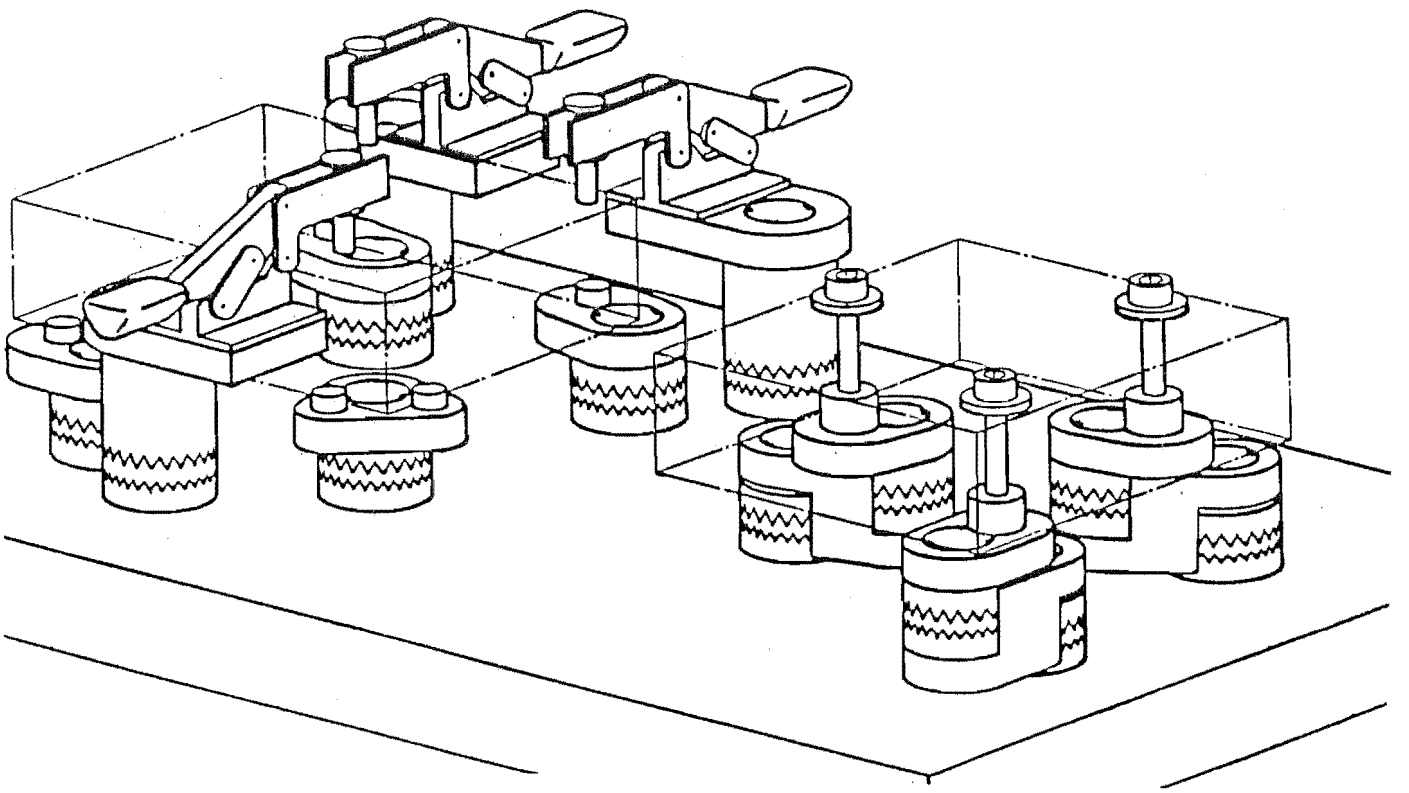


Figure 66 Special-washer fixture
(from J R Woodward and D Graham [34])

Automatically reconfigured fixture [40]

Asada and By, uses a magnetic baseplate, instead of the bolts and nut concept (figure 67). Using magnetic baseplate eliminates the problem of fastening the fixture element onto the baseplate, however, it restricts its application to machining operations involving low forces. The elements are designed for easy and precise handling as required by the specification of robotic assembly. A set of ready-made locators, clamps, and supports are kept on a magazine, the robotic arm will then move these elements from the magazine and place it precisely on the magnetic baseplate.

Advantages

1 The construction of the fixture is simple and can be done automatically.

2 The fixture does not have complicated mechanisms and is therefore reliable.

Disadvantages

1 It is not as versatile as conventional modular fixture.

2 It is only suitable for low force applications

Fluidized-baseplate fixture

A type of fixture developed early in this research is shown in figure 68. The proposed fluidized-baseplate concept, uses the phase change

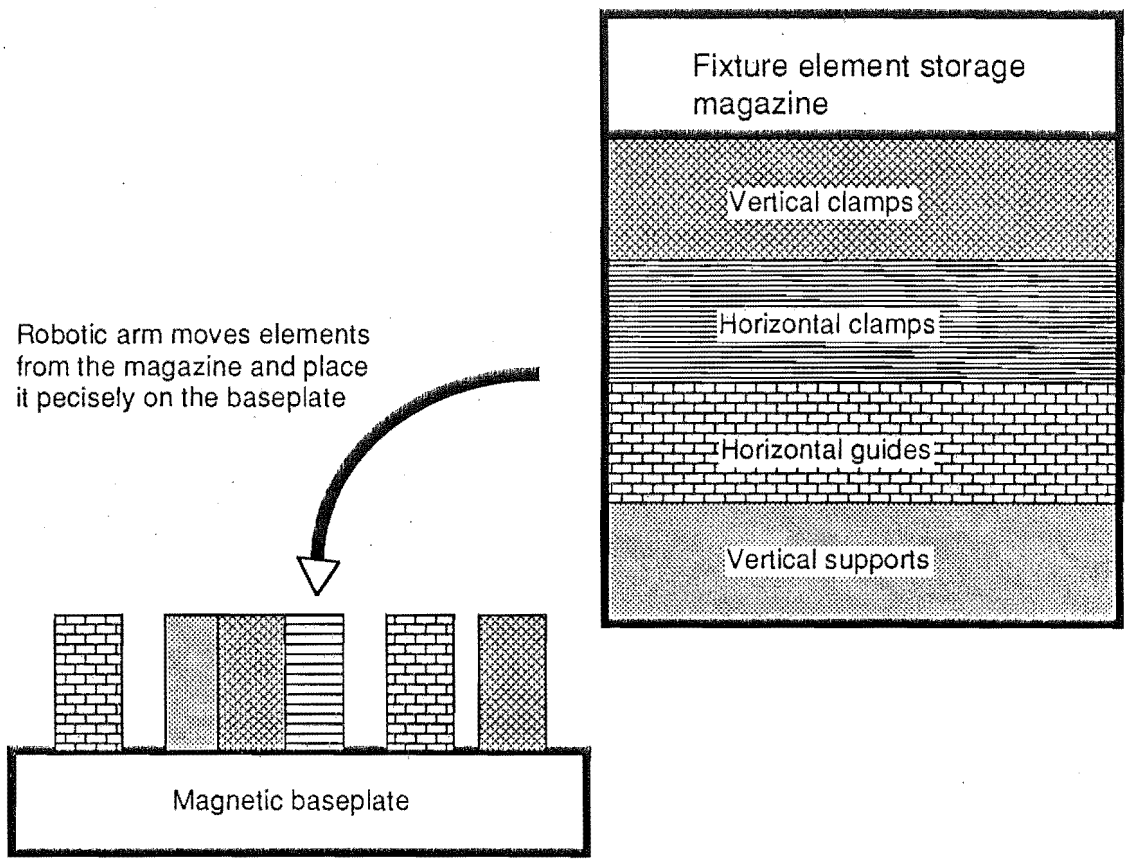


Figure 67 Automatically reconfigured clamp

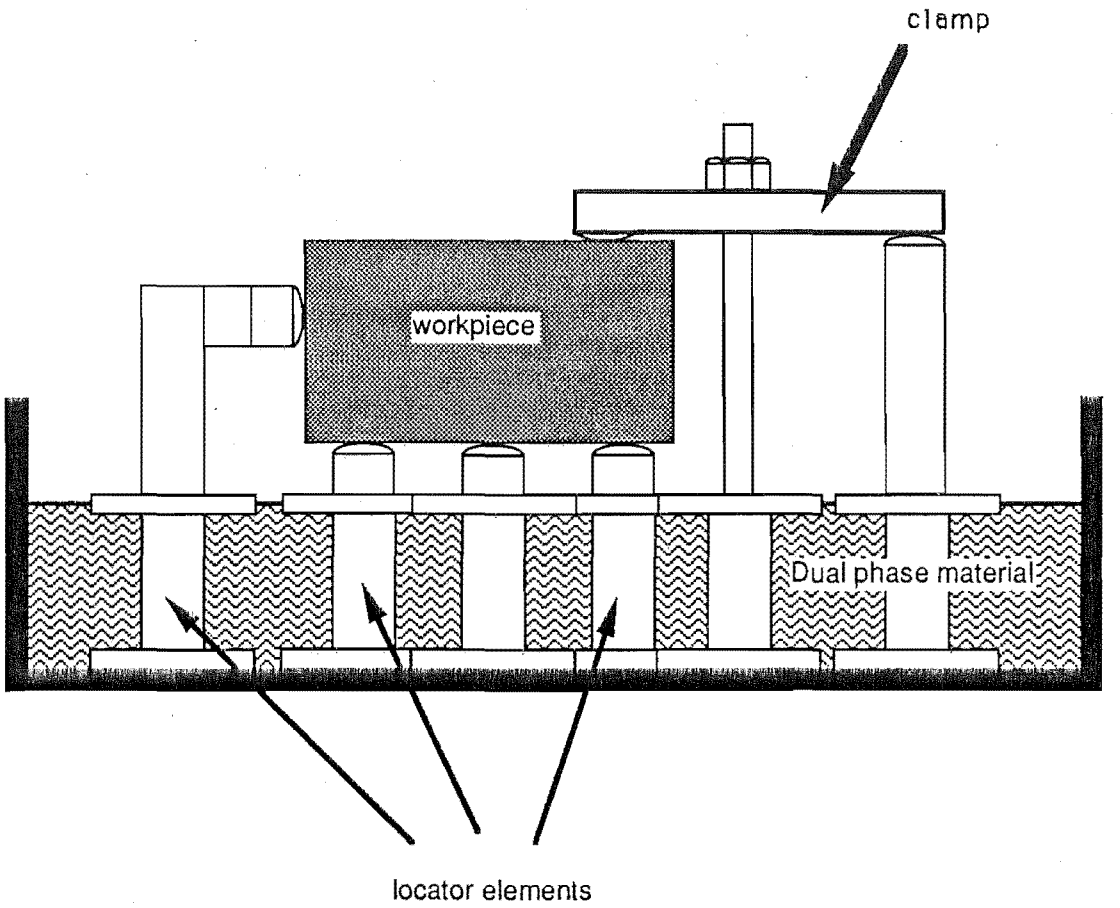


Figure 68 Fluidized baseplate concept

concept to hold the locating, clamping and support elements of a fixture in position. The dual phase material, contained in a container, functions as the body of the fixture. Since the fluidized-body cannot take high machining forces, it is suitable for low force machining operations. The advantage of this system over the fluidized-bed vise, which will be discussed in the next section, is that it does not require a separate location system to defined the workpiece position and orientation. The base of each fixturing element has to be designed to obtain maximum holding power from the fluidized body.

Advantages

1 The construction process is simple.

2 The fixture is relatively reliable and cheap to operate.

Disadvantages

1 It is not as versatile as conventional modular fixture.

2 It is suitable for low force applications

3 "Sculptured-surface" fixtures/workholding device

A different technique of fixturing/workholding is required for workpieces with sculptured surfaces; for example, a turbine blade. This could be realised in several ways. One way is to have a bed of plungers which can retract to variable depths to conform to the contour of the workpiece. Another is to use the dual phase fixturing materials. Dual phase fixturing concept uses a certain class of material

where the phase change can be controlled easily. The various methods to hold workpieces with sculptured surfaces are outlined in this section.

Programmable conformable clamp [38,35]

The programmable conformable clamp, developed at Carnegie-Mellon University, is suitable for holding compressor and turbine blades for machining (Figure 69). It employs movable plungers and a high strength belt to conform to the profile of the blade. The plungers are then locked rigidly in place.

The clamps consist of octagonal frames that are hinged so that they may be opened to accept a blade and then be closed. Usually two or three clamps are placed along the length of the blade to hold it rigidly for machining purposes.

The lower half of each clamp employs plungers that, when released, are free to conform to the profile of the blade. Before activating the plunger, the high strength belt is wrapped over the convex surface of the blade to support the blade against the plungers. These plungers are forced against the surface of the blade using air pressure. With the plungers pressed against the blade, the plungers are mechanically locked in place by locking screws and the air supply disconnected.

Advantages

1 The clamp can conform to sculptured surfaces which makes it ideal for holding turbine or compressor blades.

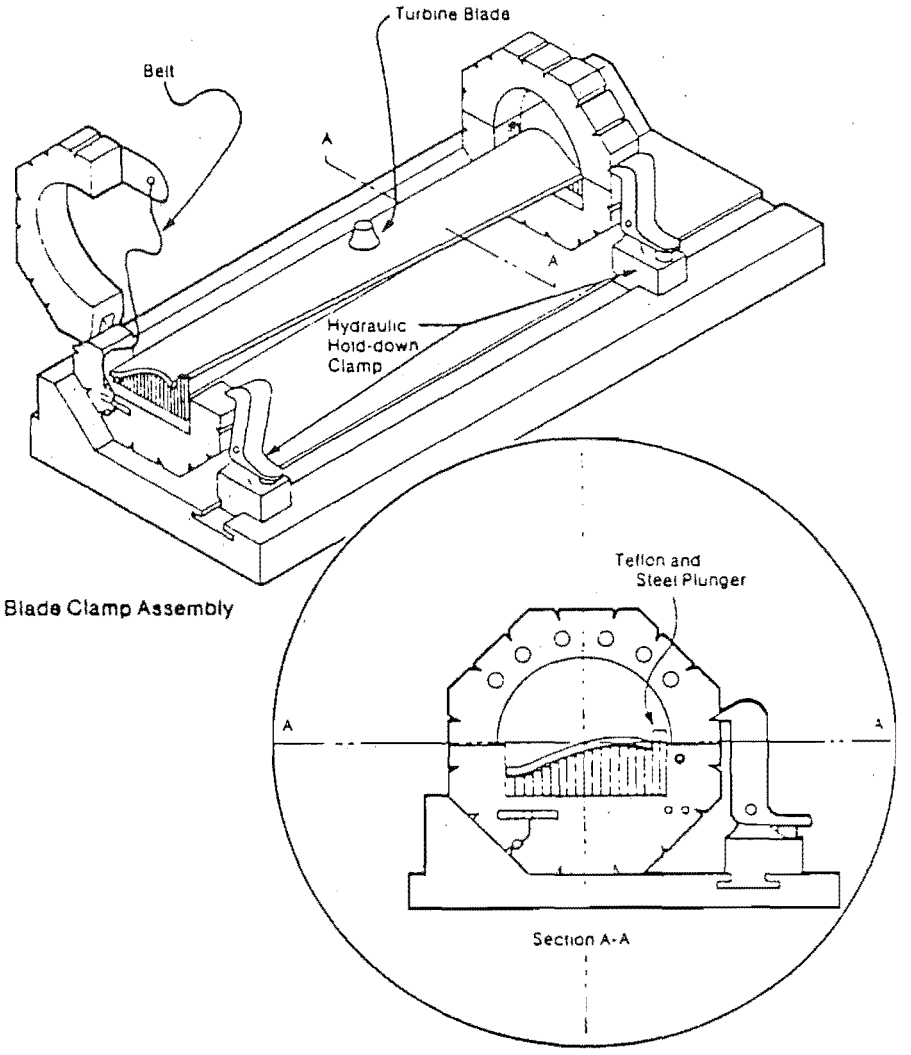


Figure 69 Programmable conformable clamp (from B S Thompson [35])

2 It requires a relatively short fixture design and manufacture lead time.

3 The clamps are compact and light enough to travel with the workpiece.

Disadvantages

1 There are a large number of moving parts, therefore, it is vulnerable to jams and damages unless chips are prevented from reaching the plungers. Repair can be quite expensive in view of the delicate nature of the clamp.

2 The clamp is specially designed for holding turbine and compressor blades; it holds the blade and allows the root to be machined. Very few other parts are of such configuration, therefore its application is quite restricted.

Encapsulation [35]

This approach employs dual phase fixturing material to encapsulate the workpiece (Figure 70). The dual phase material used is a low melting point alloy. The workpiece is encapsulated, before the machining operation, so that the encapsulation does not interfere with the tool path envelope. It has been developed specifically for the milling of gas turbine and compressor blades. The capsule that encapsulates the blade protects the delicate curvilinear airfoil section of the blade, against any damages during the machining operation.

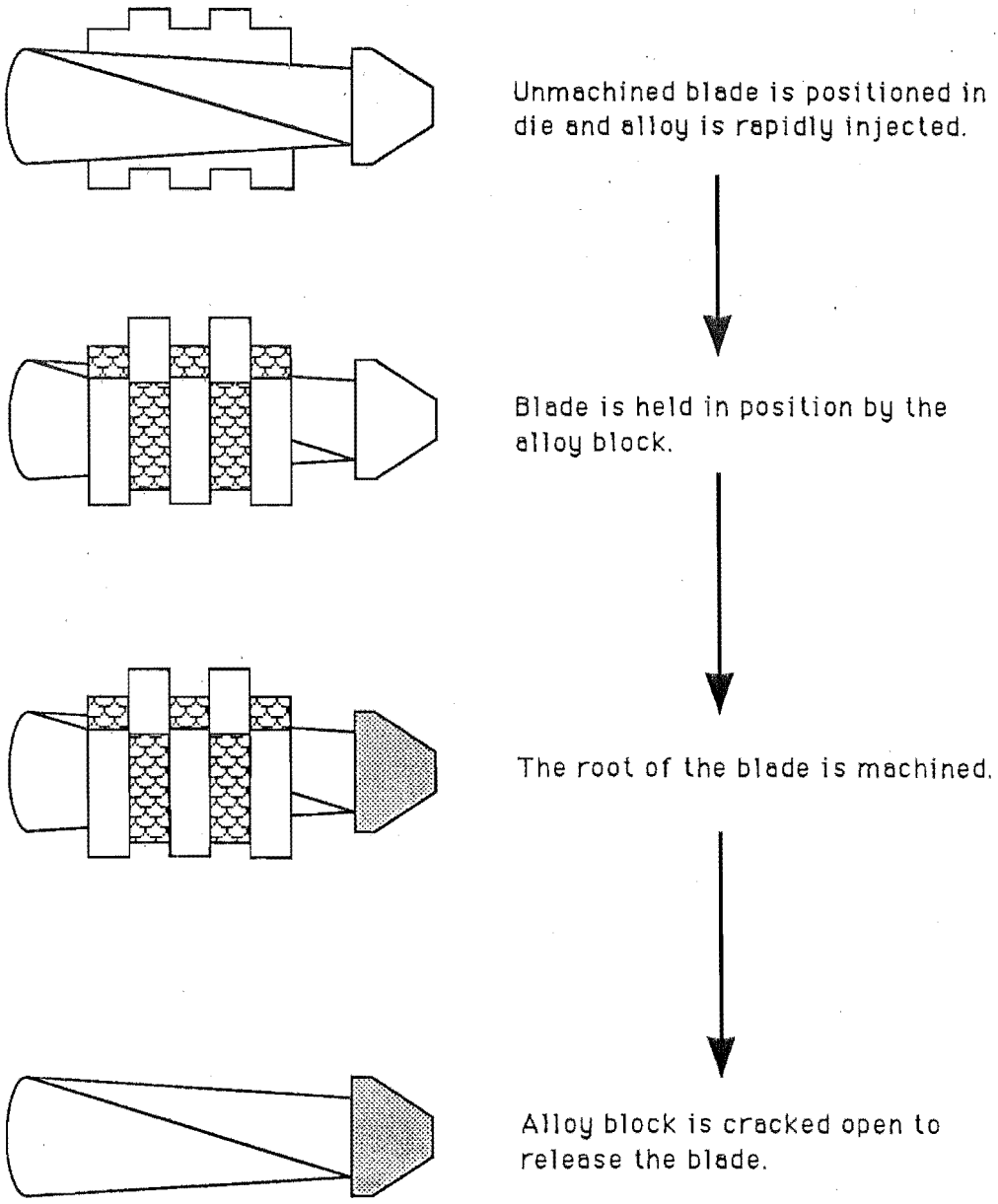


Figure 70 Encapsulation

The unmachined blade is initially positioned in an injected metal encapsulation machine. Then the dual phase material, in a liquid state, is injected into the cavity in the tool that surrounds the blade. When the dual phase material solidifies it forms a capsule which holds the blade precisely in position relative to the capsule surface. The encapsulated blade is then ejected, ready for its machining operation. When the machining operation is completed, the capsule is cracked open by a separation device and the alloy is re-used.

Advantages

1 The low melting point alloy conforms to the shape of any geometrical surface which makes it suitable for holding turbine or compressor blades.

2 The method accurately locates the workpiece, therefore no further positioning of the workpiece is required.

3 The lead time required for encapsulating the workpiece is short and the encapsulation is performed automatically.

Disadvantages

1 The machines used to perform the encapsulation are extremely expensive.

2 There is a workpiece size limitation of 300 mm (13 inches). This method is specially designed for holding turbine and compressor blades; it holds the blade and allows the root to be

machined. Very few other parts are of such configuration, therefore its application is quite restricted.

3 The die must accurately position the workpiece prior to encapsulation. Hence a different die is required for each workpiece. This would be quite expensive and the time required to produce the die does not meet the flexibility requirements of a flexible fixturing system.

Fluidized-bed vise [37,39]

The fluidized-bed concept also applies the dual phase fixturing material to hold the workpiece. A schematic diagram of the fixture is shown in Figure 71 which shows a container filled with the dual phase material.

The workpiece is immersed into a dual phase material when the material is in fluid phase. By altering certain conditions, the fluid is changed to a solid. The solidified material holds the component during different manufacturing processes, and is then made to change back into a fluid to release the component. Unlike the encapsulation and the programmable conformable clamp, which is specifically design for holding turbine and compressor blade, the fluidized-bed vise is suitable for general purpose machining operations involving light cutting forces.

Advantages

1 This method allows a wide variety of different part geometries to be clamped because a liquid is the ultimate conformable surface.

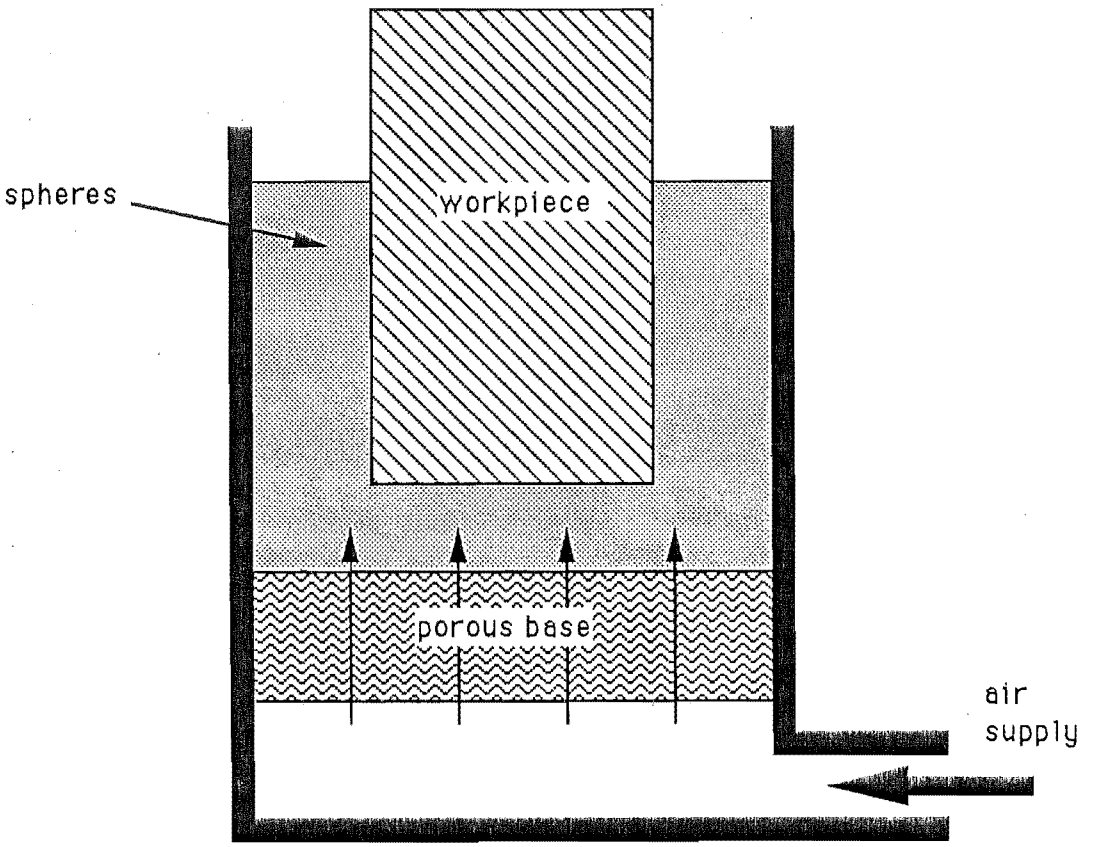


Figure 71 Fluidized bed vise

2 The device is does not have very delicate and complicated mechanism, therefore it should be relatively reliable and cheap to operate.

3 The fluidized-bed vise can hold parts without any fixture design and manufacture lead time.

Disadvantages

1 Restricted to machining operations involving light cutting forces.

2 This concept needs a separate location system to defined the workpiece position and orientation.

APPENDIX B

TURBO PASCAL MEMORY MAP

This section is extracted from the turbo pascal 4.0 owner's handbook [63]. Figure 72 shows the memory map of turbo pascal program.

The Program Segment Prefix (PSP) is a 256-byte area, built by MS-DOS when the EXE file is loaded. The segment address of PSP is stored in the pre-declared word variable PrefixSeg.

Each module, which includes the main program and each unit, has its own code segment. The main program occupies the first code segment; the code segments that follow it are occupied by the units (in reverse order from how they are listed in the uses clause), and the lst code segment is occupied by the runtime library (the System unit). The size of a single code segment cannot exceed 64K, but the total size of the code is limited only by the available memory.

The data segment (addressed through DS) contains all typed constants followed by all global variables. The DS register is never changed during program execution. The size of the data segment cannot exceed 64K.

On entry to the program, the stack segment register (SS) and the stack pointer (SP) are loaded so that SS:SP points to the first byte past the stack segment. The SS register is never changed during program execution, but SP can move downward until it reaches the bottom of the segment. The size of the stack segment cannot exceed 64K; the

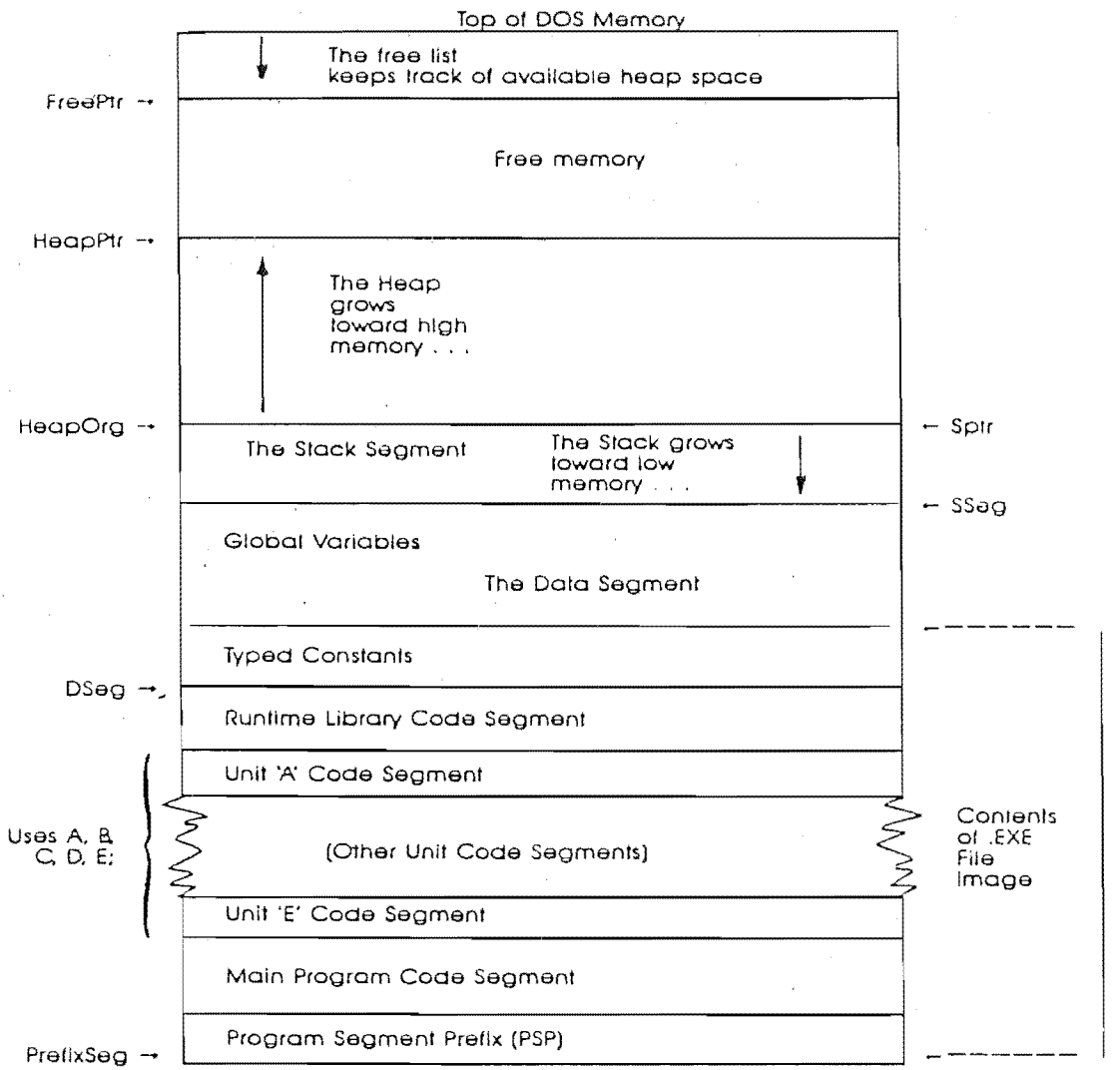


Figure 72 Memory map of Turbo Pascal program (from "Turbo Pascal" Owner's Handbook [63])

default size is 16K, but this can be changed with a $M compiler directive.

The heap stores dynamic variables, that is, variables allocated through calls to the New and GetMem standard procedures. It occupies all or some of the free memory left when a program is executed. The actual size of the heap depends on the minimum and maximum heap values, which can be set with the $M compiler directive. Its size is guaranteed to be at least the minimum heap size and never more than the maximum heap size. If the minimum amount of memory is not available, the program would not execute. The default heap minimum is 0 bytes, and the default heap maximum is 640 Kb; this means that by default, the heap will occupy all remaining memory.

The heap is a stack-like structure that grows from low memory in the heap segment. The bottom of the heap is stored in the variable HeapOrg, and the top of the heap, corresponding to the bottom of free memory, is stored in the variable HeapPtr. Each time a dynamic variable is allocated on the heap (via new or GetMem), the heap manager moves HeapPtr upward by the size of the variable, in effect stacking the dynamic variables on top of each other.

The maximum size of a single variable that can be allocated on the heap is 65521 bytes, since every variable must be completely contained in a single segment.